UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Aplicación de diseño y seguimiento de rutas a pie

Autor: Javier de Marco Tomás

Tutor: Carlos Aguirre Maeso

Dpto. Ingeniería Informática

Escuela Politécnica Superior

Universidad Autónoma de Madrid

MAYO 2020

Resumen (Castellano)

Este Trabajo de Fin de Grado se centra en el desarrollo e implementación de una aplicación GPS para el sistema Android, en el cual se podrá diseñar y utilizar rutas habituales a pie del usuario.

Se trata de dar las herramientas necesarias para la creación y posterior guardado de los diferentes itinerarios que el usuario considere recurrentes o interesantes para su uso frecuente.

El software que se ha desarrollado no pretende ser una sustitución a lo ya existente, sino más bien una alternativa útil que aporte una combinación de uso tanto en ciudad como en montaña, para aquellos trayectos que el usuario haya diseñado por su propia cuenta con los instrumentos que la aplicación aporta.

Se ha de aclarar que solo se ha contemplado la posibilidad de realizar las rutas a pie, por lo que otras opciones no están incluidas. Dicho esto, la aportación clara del software es la posibilidad de realizar una navegación guiada por varios puntos, siendo estos considerados puntos A-B o etapas, en un circuito que puede contener muchas etapas.

Existen varias posibilidades para la invención de trayectos. La más sencilla es utilizar el propio mapa para pulsar en los diversos puntos por los que se quiere parsear en la ruta, creando así una sucesión de puntos que se pueden visualizar en la pantalla. Otra posibilidad es utilizar el botón de búsqueda que se incluye para encontrar un punto de interés en concreto que se pretende añadir al circuito. Este método es parecido a cualquier buscador de puntos de interés de una aplicación GPS: utiliza una base de datos para encontrar lo más parecido a lo introducido en el cuadro de texto, presentando una serie de resultados que el usuario puede escoger, si alguno de ellos se corresponde con lo que se espera. Tras la selección, el mapa nos llevará hasta dicho punto, creando una nueva etapa desde el último punto utilizado hasta el buscado. Existe la posibilidad de mezclar y utilizar los dos métodos simultáneamente, creando algunos puntos a mano y otros con el explorador de puntos de interés. También, en el caso en el que el usuario se equivoque, la aplicación cuenta con un sistema para deshacer el último punto añadido.

Por otro lado, la aplicación cuenta con la posibilidad de gestionar puntos habituales del usuario y generar con ellos rutas habituales más rápidamente.

Utilizando el mismo buscador de puntos se podrá añadir puntos habituales a una lista independiente, la cual posteriormente es posible visualizar. Estos puntos actúan como puntos habituales y se podrán ver, eliminar y añadir. Si se utilizan estos puntos, se podrá generar rutas habituales y guardarlas en el menú principal para su posterior uso. Estos circuitos tendrán un nombre estándar que contendrá la unión de los nombres de los puntos habituales.

La aplicación se ha considerado solo para los smartphones con el sistema operativo Android, y se ha desarrollado con el framework que Android Studio ofrece, por lo que el lenguaje utilizado para su programación es Java.

Abstract (English)

This essay is centered around the development of a GPS Android software in which the user will have the ability to create and design foot routes which are common in his day to day routine.

This Application will try to give the user the necessary tools to create, design and save trajectories which he thinks are recurrent so they can be use daily.

This software does not try to be a substitute of what the user uses as a GPS Application, but more of a good alternative that combines city and mountain usage for routes that the user creates with the tools that the APP gives him.

It is worth saying that this GPS software only takes to account foot routes, so any other option has not been implemented. Once this is said, what this software gives the user is the possibility of doing guided foot trajectories between various points, which they are considered A-B, or stages on a multipoint circuit.

There are two ways of creating trajectories. The simplest one is by using the user's finger to click in the map the points from which he wishes to pass through, making a route connection each point in succession which can be seen in the screen. The other way is to use the search functionality to find points of interest that the user wants to add to the route. This method is similar to other point of interest search engines of other GPS Applications, it uses a data base

to find the result that suits the text that has been introduce and present a series of results from which the user can choose from if any of them is the one searched. After we select one, the map will move to that point making a stage between the last point added and the one searched. There is the possibility to combine and use both methods simultaneously, creating points with the manual method or by searching points of interest. Additionally, if the user makes a mistake there is an option to undo the last point added to the route.

Aside from that, it is possible to manage habitual points searched by the user and generate habitual routes faster. Using the same search engine as before the user can add habitual points to a separate list where they can be seen and eliminated if necessary. With these points it is possible to create a route adding it to the main menu for recurrent use. The Application will automatically generate a name for this route with the names of the habitual points use to create it

The Application only works on smartphones with the Android operating system, and it has been develop using Android Studio Framework, consequently the language used is Java.

Palabras clave (castellano)

Android, Android Studio, Framework, GPS, navegación, ruta, rutas a pie.

Keywords (inglés)

Android, Android Studio, Framework, GPS, navigation, route, foot routes.

Agradecimientos

A mi padre, por apoyarme incondicionalmente tras cada error y porque cierro una etapa de la que él estaría orgullo.

A mi madre, por su amor y cariño incondicional.

Y a mi pareja, Pilar, que me dio la alegría y la fuerza para continuar y acabar.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

1 Introducción

1.1 Motivación

Hoy en día la utilización de la tecnología GPS y la navegación se ha convertido en una herramienta esencial a la hora de movernos. Es muy común utilizar estos sistemas en momentos en los que no conocemos de manera precisa la ciudad o lugar donde nos encontramos, y recurrimos a aplicaciones que nos indican el camino óptimo desde un punto inicial a un punto final. Muchas aplicaciones únicamente permiten la creación de caminos entre dos puntos, y las que permiten rutas con más de dos puntos no dan la posibilidad de guardarlas, por lo que la gestión de estas es inexistente.

También es común su uso fuera de las poblaciones, donde únicamente existen caminos o senderos, los cuales se recorren a pie realizando actividades de senderismo. Estas aplicaciones permiten la generación de rutas y su guardado, pero su uso está plenamente ajustado a las actividades de montaña y al aire libre, por lo que un usuario de ciudad encontraría inútil este tipo de software.

Con este TFG se intenta dar una solución amplia que permita a los usuarios diseñar, crear, guardar y utilizar las rutas que desee, tanto de ciudad como de montaña, dando un mapa mixto que incluye información de ambas y dando una herramienta para diseñar y gestionar circuitos a pie.

También se ha tenido en cuenta que el usuario pueda utilizar estas rutas creadas de forma recurrente, por lo que se han incluido sistemas de guardado y salvado de rutas.

En muchas ocasiones nos encontramos con aplicaciones que, por incluir numerosas funciones y características, dificultan su utilización, dejando de lado una mentalidad sencilla y práctica a la hora de diseñar un software. Por lo que, en muchas ocasiones, personas con dificultad tecnológica, por su edad o desconocimiento del uso de aplicaciones, sufren al no poder utilizar este tipo de software, dejando de lado el gran beneficio que estas podrían aportarles. Por ello, es necesario crear software sencillos de manejar, con unas características y funcionalidades estrictas, que sean potentes pero no difíciles, y que reduzcan la brecha tecnológica que algunas personas puedan padecer.

1.2 Objetivos

Esta aplicación se creó con un objetivo claro, ser una herramienta sencilla de utilizar, mediante la cual una persona pueda crear, gestionar y utilizar las diferentes rutas que crea convenientes y que sean de uso frecuente. También podrá utilizar las funciones de navegación para llegar desde el punto de inicio al punto final de la manera más rápida, pasando por los puntos intermedios dictados por el mismo.

Uno de los objetivos principales del software aquí descrito es el diseño de circuitos a pie. Se entiende por esto la posibilidad de que, en una interfaz gráfica, un usuario pueda buscar y añadir puntos, que se enlazarán según su momento de agregación. Estos puntos podrán ser añadidos de dos formas. La forma más sencilla es la simple pulsación en cualquier lugar del mapa. Como forma alternativa, se podrá utilizar el buscador para agregar puntos de interés de una forma rápida y sencilla, cuando se conozca aquello que se busca. Si en cualquier momento el usuario se equivoca, existe la posibilidad de deshacer el último punto agregado. El diseño de rutas es una herramienta muy potente que permite realizar itinerarios complejos, ajustados a la medida y deseo del usuario.

Una vez diseñada la serie de puntos, la aplicación dará la posibilidad de realizar un salvado con un nombre específico, para mayor comodidad y facilidad de gestión de las numerosas rutas que el usuario pueda tener. Esta función es una de las piezas clave de la aplicación, ya que permite al usuario mantener y conservar de manera sencilla una colección de circuitos. Se trata de una característica importantísima debido a la enorme utilidad que proporciona la opción de guardar una ruta, por compleja que sea, diseñada y creada por el usuario.

Es importante darle finalidad y uso a los itinerarios que el usuario haya diseñado y creado, ya que así se fomenta el uso de la aplicación y aumenta considerablemente el atractivo de esta. Por ello, se ha introducido como objetivo la creación de un sistema de navegación que permita al usuario recorrer el circuito que esté creando o haya creado, de manera sencilla y en el orden que él mismo dictaminó. Hay que tener en cuenta esta característica, el orden de los puntos, ya que es así como el usuario lo pensó y desea. Por ello el sistema guiará al usuario desde el punto inicial al final, pasando por aquellos puntos intermedios y en el orden que hayan sido añadidos a la ruta, utilizando siempre el camino más corto entre puntos, optimizando así el tiempo y la distancia.

Como objetivo final, se propone la creación y guardado de puntos habituales. Estos puntos son aquellos que el usuario va a utilizar con frecuencia, por lo que es muy útil tener la posibilidad de manejar este tipo de puntos. La aplicación dará la posibilidad de crear rutas habituales, que son rutas compuestas por puntos habituales. Estos circuitos se guardarán en la memoria. interna del dispositivo, por lo que el usuario podrá conservar las rutas una vez cierre la aplicación.

Al mencionar toda la serie de objetivos, se ha recalcado el propósito de que sea una aplicación sencilla de usar, pero potente en sus funcionalidades. Se ha tomado como objetivo la realización de un sistema con unas características y funcionalidades completas y atractivas, y diseñadas de manera que el usuario final a la hora de utilizar el software encuentre fácil la utilización de todas sus cualidades y pueda sacar el mayor provecho posible sin tener un conocimiento amplio de la aplicación. Es importante tener este fin presente en todas las etapas de diseño y desarrollo para mantener en alta prioridad el uso sencillo del software.

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos:

- **Estado del arte:** En esta sección se expone la actualidad relacionada con los navegadores, haciendo hincapié en la navegación a pie. Se da, a continuación, una visión crítica de la misma y se argumenta el valor añadido de este proyecto.

- **Diseño:** Se presentan los diagramas y decisiones de diseño del software desarrollado junto con una explicación clara y concisa sobre dichos temas.

- **Desarrollo:** Dado el diseño en el apartado anterior, se expone la aplicación práctica de dicho diseño y cómo se ha implementado cada uno de los componentes descritos.

- **Integración, pruebas y resultados:** Acabado el desarrollo, es necesaria la realización de pruebas para comprobar de la mejor forma posible el correcto funcionamiento del software.

- **Conclusión:** Tras las pruebas, se exponen las conclusiones alcanzadas de lo aportado a las tecnologías actuales y de los objetivos descritos con anterioridad.

2 Estado del arte

En la actualidad, existen numerosas aplicaciones en multitud de plataformas con la funcionalidad GPS para guiar al usuario desde su localización hasta su destino.

La primera es Google Maps, un software administrado y soportado por la compañía Google. Aunque originalmente era una aplicación web lanzada en el año 2004, no fue hasta el año 2008 cuando Google creó la versión para Android de esta aplicación.

Esta aplicación tiene una gran base de datos de puntos de interés, comercios y lugares, lo que hace su uso muy sencillo a la hora de buscar cualquier lugar al que se desee llegar. Se trata de una aplicación de mapas y funciones de navegación que permite a los usuarios buscar, crear y usar rutas. Tiene la posibilidad de guardar lugares favoritos y utilizarlos para crear rutas con ellos. Ofrece navegación en multitud de formas, a pie, en coche, en bicicleta, en transporte público. También cuenta con diferentes tipos de mapas, tales como imagen de satélite, mapa topográfico y una opción con una estética más minimalista.

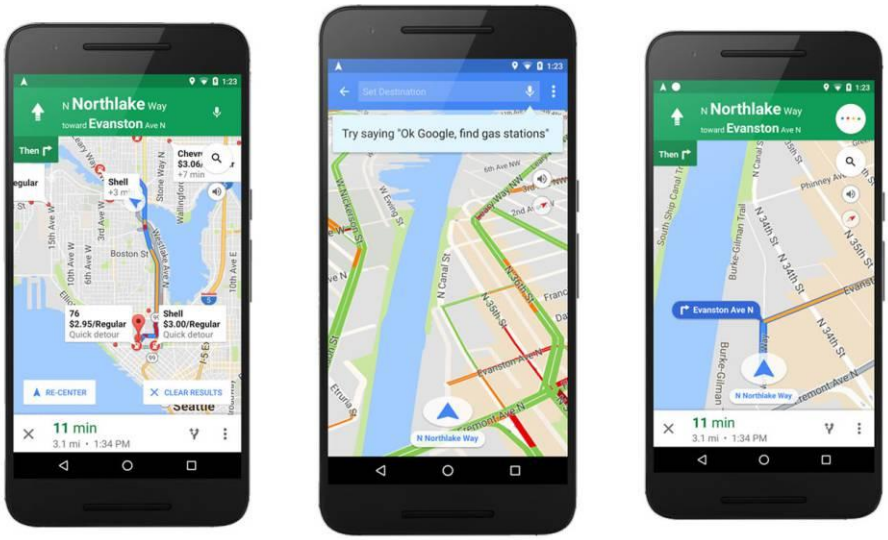

Figura 1: Google Maps UI

La siguiente se llama Sygic, y fue fundada en 2004. Su aplicación Android fue la segunda en soportar este tipo de sistemas. Cuenta tanto con navegación a pie como en coche, con instrucciones curva a curva. Su gestión de rutas es parecida a la de Google Maps. Tiene la posibilidad de guardar puntos de interés para, posteriormente, crear una ruta. Contiene también una enorme base de datos para la búsqueda de puntos de interés.

Figura 2: Sygic UI

Como tercera aplicación está TomTom. Esta aplicación cuenta con un sistema de navegación curva a curva, desde un origen a un destino. Es posible guardar puntos como favoritos para su uso posterior en rutas.

Figura 3: TomTom UI

El uso más recurrente de estas tres aplicaciones anteriores es el de navegación de rutas urbanas o interurbanas. Dentro del sistema Android han surgido otro tipo de navegadores de rutas más orientadas a la montaña y al senderismo. Estas aplicaciones se centran en rutas a pie de multipunto y suelen contar con un mapa topográfico.

Entre estas aplicaciones, se encuentran:

ViewRanger, un software dedicado a las rutas de montaña a pie generadas por sus usuarios. En esta aplicación tenemos una navegación punto a punto basado en trazos creados por personas que utilizan la aplicación. Podremos seleccionar la ruta deseada y recorrerla ayudados por el sistema GPS del móvil. Aquí se permite la creación de itinerarios punto a punto clickeando en el mapa aquellos lugares por los que se quiere pasar. Es posible conservar la ruta guardando los puntos seleccionados en favoritos con un nombre para su uso recurrente. A diferencia de las aplicaciones GPS urbanas, esta nos da el tiempo que ha transcurrido, no el de llegada, por lo que su finalidad es más la realización de una actividad física que la navegación por el camino más corto a un lugar de interés.

Figura 4: ViewRanger UI

A diferencia de las urbanas, este tipo de aplicación se centra en las rutas a pie, pero en un formato destinado al senderismo.

Se puede ver que en las aplicaciones convencionales GPS se centra la atención en el uso de puntos y de la navegación entre ellos por los diferentes métodos que existen.

Este tipo de aplicaciones centran su objetivo en encontrar el camino que se quiere recorrer, y la forma de acceso a rutas multipunto se realiza de forma compleja. Pero si nos centramos en su uso cotidiano, puede resultar poco intuitiva la forma de gestionar estas rutas.

Por ello se propone un sistema de diseño y gestión de rutas a pie que permita al usuario manejarlo de forma sencilla. Si damos importancia a las rutas en vez de a los puntos, se crea una aplicación que se destina a los itinerarios habituales de un usuario, teniendo la posibilidad de seleccionar el mismo trayecto de una forma rápida y sencilla.

Debido también a la importancia de puntos habituales que pueda tener el usuario, se ha incluido la gestión de estos para la posterior creación de rutas habituales.

3 Diseño

Uno de los conceptos más importantes a la hora de desarrollar software es el diseño. Es en este momento en el que se toman las decisiones que posteriormente se materializarán en la implementación para dar vida a la aplicación. Debido a la importancia de realizar un diseño apropiado, se utilizó el lenguaje UML para describir los distintos diagramas, utilizando la herramienta online de Draw.io.

3.1 Diseño de Módulos

Para comenzar, es de gran relevancia en el diseño de cualquier tipo de software separar en módulos la funcionalidad de nuestra aplicación:

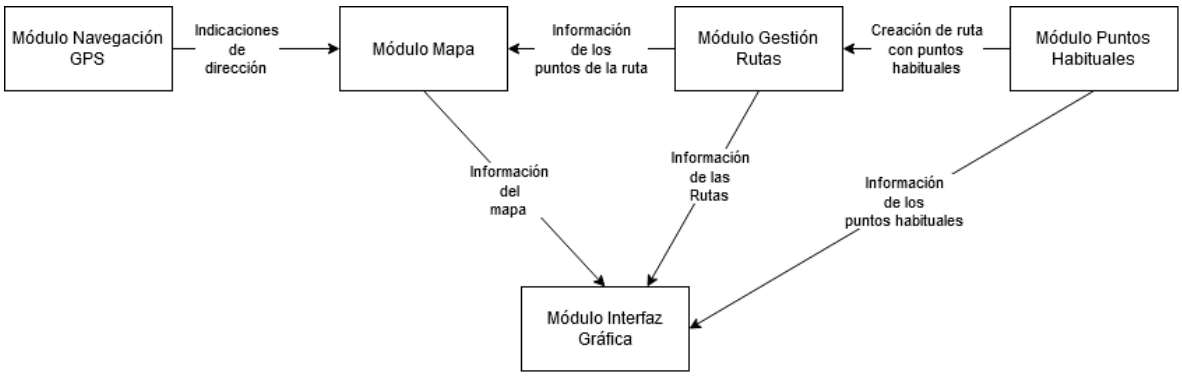

Figura 5: Diseño módulos

Se puede ver claramente que existen cinco módulos independientes que se conectan y envían información entre sí para llevar a cabo las labores de la aplicación:

Primeramente, el módulo de navegación GPS, el cual gestiona la localización del usuario y la dirección que este tiene que tomar en cada momento y envía la información al módulo del mapa.

El módulo del mapa recibe estas indicaciones, además de posibles puntos o rutas del gestor de rutas, y, a su vez, remite todo esto al módulo de la interfaz gráfica.

La interfaz gráfica es, en nuestro caso, la encargada de mostrar toda la información relevante al mapa para que el usuario pueda visualizar el camino a seguir y, a su vez, la encargada de mostrar, en un menú principal, todas las rutas guardadas por el usuario con anterioridad.

El módulo de gestión de rutas guardará la información relevante a los itinerarios que el usuario considere. Además, mandará la información pertinente a la interfaz gráfica para que esta la muestre en un menú principal; o al mapa, para mostrar los puntos relevantes de la misma en él.

Por último, el módulo de puntos habituales, que se encargará de gestionar este tipo de puntos. Su cometido es el de crear, visualizar y eliminar puntos. Estos puntos creados se enviarán con la información pertinente al módulo de rutas para que este cree la ruta con dichos puntos.

3.2 Diseño de Procesos

Al diseñar cualquier componente de una aplicación software es importante conocer los procesos, usos y operaciones que va a soportar. Por ello, se han descrito diagramas de estados y casos de uso, que definen de manera visual y sencilla de entender la forma en la que se comportará el flujo del software.

Primeramente, se realizó el diagrama de estados, que define los posibles puntos por los que una aplicación software puede encontrarse, desde su punto de entrada a su terminación. Esto nos permite delimitar un marco funcional que guiará cualquier paso futuro de diseño y desarrollo.

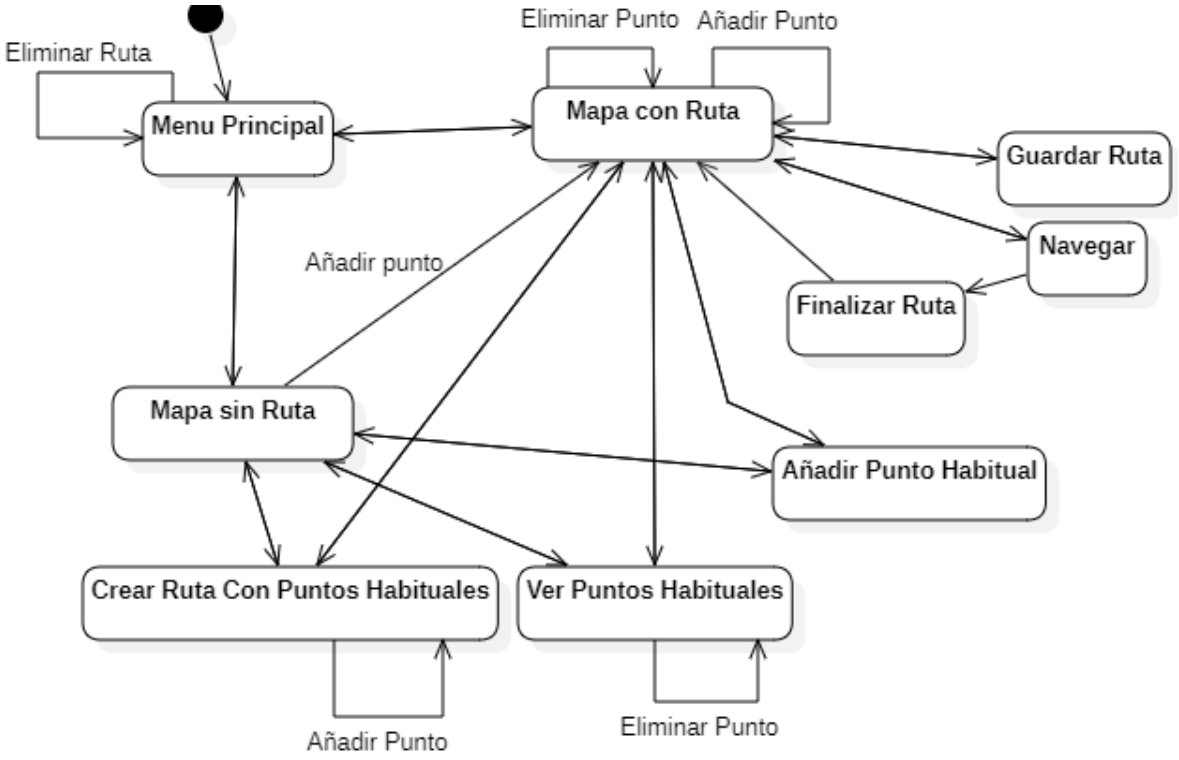

Figura 6: Diagrama de estados

El punto de entrada en la aplicación es el Menú Principal, en el cual se verán las rutas que el usuario haya guardado. La primera vez que se inicie, se pedirá confirmación para acceder a la localización del usuario y así poder utilizar las funciones de GPS. En este Menú Principal podremos, una vez se hayan guardado rutas, eliminar los circuitos que ya no utilicemos. El proceso de eliminación es sencillo. Si se realiza un pulsado largo en el nombre de cualquier ruta, se mostrará una confirmación de eliminado. Si aceptamos, tendremos que pulsar en el botón de actualizar situado en el extremo inferior izquierdo para que la lista de rutas se actualice con el eliminado realizado.

Si, por el contrario, lo que se quiere es acceder al mapa, existen dos posibilidades:

La primera es iniciar el mapa seleccionando un botón destinado a ello, con lo que la aplicación pasará al estado de Mapa sin Ruta, donde se nos mostrará la localización del usuario sin ninguna ruta cargada. La segunda posibilidad es la de seleccionar una ruta de la lista de rutas que se muestran. Esto activará el mapa igualmente, con nuestra localización, pero esta vez con la ruta cargada que haya sido seleccionada.

Para continuar, y por simplificar, se hablará del mapa con y sin ruta, por igual, ya que en términos de funcionalidad es igual, aunque en términos de estado no lo sea.

En la actividad de mapa es donde se podrán realizar las operaciones más complejas de la aplicación. Primeramente, y la más sencilla, es añadir puntos. Esta operación se puede realizar de dos maneras posibles. Si pinchamos con el dedo en la pantalla en cualquier lugar del mapa, se mostrará un punto en ese mismo lugar. Eso nos indica que hemos agregado un punto con la función táctil del mapa. Si, por el contrario, queremos buscar un lugar en concreto, existe un botón en la parte inferior derecha con forma de lupa que, al pulsarlo, nos mostrará el buscador de lugares de interés. Este buscador reaccionará según se introduzcan caracteres en el lugar del cuadro de texto, y se indicarán los lugares que tengan similitud con lo introducido. Una vez escogido el lugar, la aplicación acercará la cámara al lugar aproximado del punto y añadirá dicho punto a la ruta, ya sea el primero o los sucesivos, creando así una ruta. Estos dos métodos se pueden utilizar libremente entre sí, ya que la aplicación se encargará de gestionar la ruta según vayamos agregando puntos.

Si en cualquier momento sentimos que nos hemos equivocado y deseamos eliminar un punto, existe un botón destinado a ello. Al pulsarlo se eliminará el último punto añadido de la ruta. Esta operación se podrá utilizar hasta eliminar el último punto de la ruta, eliminando así del mapa la ruta por completo.

Existen dos operaciones claras con respecto a un circuito cargado en el mapa: la posibilidad de guardar la ruta que está mostrando el mapa y la de navegar por la misma. Si lo que deseamos es guardar, existe un botón indicando muy claramente la operación que realiza. Al pulsarlo se mostrará un cuadro de diálogo pidiendo al usuario un nombre para la ruta que se quiere guardar. Si el nombre de la ruta que intentamos guardar ya existe, la operación no se realizará. De lo contrario, la aplicación guardará esta ruta en un fichero en la memoria interna del teléfono móvil, para su muestreo y uso posterior en el Menú Principal.

Una vez creada y guardada una ruta, la siguiente operación de interés es la de navegar por la misma. La aplicación cuenta con funciones de navegación que guía al usuario desde su localización hasta el destino (último punto de la ruta), pasando por todos los puntos intermedios. Es notable mencionar que, debido a fines estéticos, se ha optado por mostrar la ruta sin tener en cuenta la localización del usuario. Únicamente se tiene en cuenta una vez este haya pulsado el botón de navegar, tras lo cual la aplicación genera un nuevo circuito, incluyendo como origen la localización del usuario. La navegación cuenta con instrucciones de voz e información relevante del tiempo y distancias restantes. Al llegar a cada punto intermedio se procederá a la navegación hasta el siguiente, y así sucesivamente hasta llegar

al destino final, tras lo cual se volverá al mapa con la ruta original cargada. El usuario tendrá la posibilidad, en cualquier momento, de cancelar la navegación y volver al mapa.

Por último, existen puntos y rutas especiales, denominadas rutas y puntos habituales. Estos puntos son aquellos que el usuario considera que su uso será muy recurrente y merece tener una forma rápida para acceder a ellos. Desde el mapa, en la parte superior derecha se podrá acceder a las funcionalidades relacionadas con este tipo especial mediante un menú desplegable. Al pulsarlo se mostrarán tres opciones: añadir, visualizar y crear ruta con puntos habituales.

Para simplificar la función de añadir punto habitual, se ha utilizado el buscador de puntos de interés. Esta función es similar a lo mencionado con anterioridad. Se trata de un buscador con un cuadro de texto, en el cual introduciremos el nombre o dirección del lugar que queremos tener como punto habitual, y la aplicación nos mostrará posibles resultados. Al pulsar uno, la cámara se moverá hasta una localización aproximada del punto escogido y saltará un cuadro de dialogo para introducir el nombre del punto habitual, el cual se guardará en la memoria interna del dispositivo.

Una vez agregados los puntos, se podrán visualizar con la opción de visualizar puntos habituales. Aquí se nos mostrará un listado de los puntos de habituales que tengamos guardados. Si ya no deseamos conservar alguno de estos punto,s se puede optar por la operación de eliminar. Si realizamos un pulsado largo sobre el nombre de uno de los puntos, se mostrará un cuadro de confirmación. Una vez confirmado, el punto será eliminado de la memoria interna del dispositivo.

Por último, se podrá crear una ruta entre estos puntos habituales. La operación se realiza de forma muy sencilla e intuitiva. Una vez se ha accedido a la actividad de crear ruta con puntos habituales, se mostrará el primer punto agregado por defecto, el cual podremos modificar pulsando sobre él. Este pulsado hace que la aplicación busque cuáles son los puntos habituales creados por el usuario y los muestre en un listado con funciones de scroll para que se seleccione el que se desee. Si se desea agregar puntos, existe un botón que sumará un punto al listado que se muestra, pudiendo modificar este también con la misma operación anterior. Se procederá de este modo hasta que se desee. Una vez acabado, se tendrá que pulsar el botón de crear ruta. Esto activará el guardado automático de la ruta, tras el cual el usuario recibirá un feedback en forma de texto en la parte inferior indicando el éxito de la

operación. El nombre de la ruta se genera automáticamente con los nombres de los puntos habituales. Esta ruta se podrá visualizar y utilizar en el Menú Principal, como si de una ruta normal se tratase.

3.3 Diseño de la Interfaz

Al tratarse de una APP para dispositivos móviles, se decidió realizar unos bocetos iniciales de la interfaz gráfica para tomar como referencia. Para la realización de estos bocetos se utilizó una herramienta online que facilitaba esta labor.

Los bocetos realizados son los siguientes:

Figura 7: Diseño menú principal

Como primer boceto tenemos el menú principal. Se trata de una interfaz minimalista, y sencilla, en la que podemos observar el nombre de la APP en la parte superior seguido de una lista de las rutas que haya guardado el usuario (si no existen, esta lista estará vacía). Por último, dos botones nos permitirán realizan dos acciones muy concretas. El primero, situado

en la parte inferior izquierda, hará la función de refrescar la lista de rutas guardadas, una vez se haya eliminado una ruta. El segundo botón, situado en la parte inferior derecha, dará la posibilidad de iniciar el mapa sin cargar ninguna ruta.

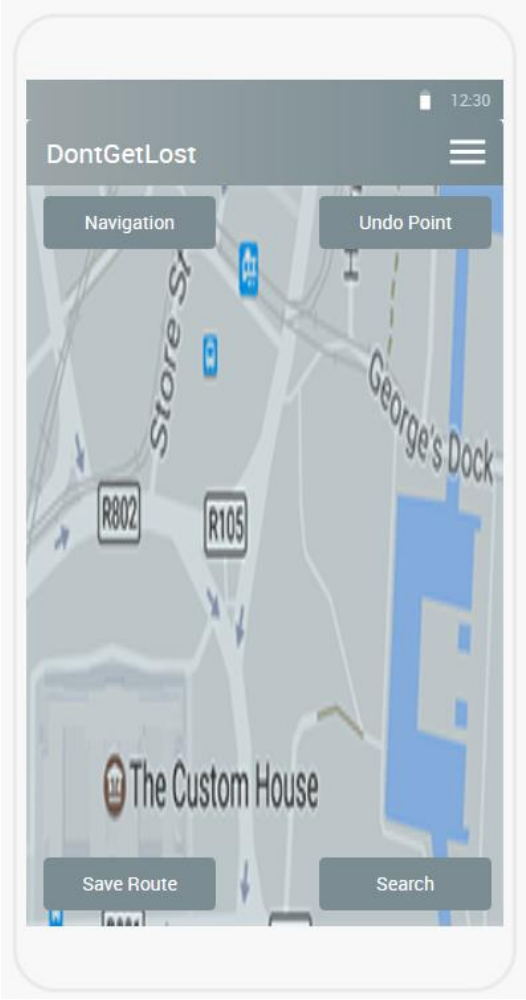

Figura 8: Diseño UI mapa

La interfaz gráfica que se ha diseñado del mapa también será muy sencilla, sin dejar de lado la funcionalidad. Primeramente, tendremos el nombre de la APP en la parte superior, seguido de un botón de menú desplegable que permitirá gestionar los puntos y rutas habituales.

Dentro ya de la interfaz del mapa, se podrá ver el mapa propiamente dicho, junto con la localización y ruta del usuario. También se podrán encontrar cuatro botones que darán funcionalidad extra, fácilmente accesibles y sin interponerse demasiado en la parte útil de la interfaz gráfica. Arriba a la izquierda se sitúa el botón de empezar a navegar una ruta, una vez haya sido creada dentro del mapa. Este botón llevará al usuario a una actividad auxiliar

que le guiará por los puntos de la ruta que haya creado. Situado en la parte superior derecha, estará el botón de deshacer punto, que dará la posibilidad al usuario de deshacer el último punto que haya introducido en el mapa.

Y situados en la parte inferior izquierda y derecha estarán los botones de guardar ruta y buscar, respectivamente. El botón de guardar accionará la aparición de un cuadro de texto para introducir el nombre de la ruta, tras lo cual guardará la ruta en la memoria interna del dispositivo. Por su parte, el botón de búsqueda lanzará una actividad auxiliar con un cuadro de texto en la parte superior, en el que se podrá introducir el punto de interés que se esté buscando. A continuación, se puede ver un ejemplo de diseño de dicha actividad auxiliar.

Figura 9: Diseño búsqueda

Tal y como se ha descrito, se tendrá un cuadro de texto en la parte superior, seguido del historial de búsquedas. A medida que el usuario introduzca texto, la aplicación mostrará posibles resultados de localizaciones a elegir y el usuario deberá escoger uno.

Una vez añadidos algunos puntos al mapa, si se pulsa el botón de guardar ruta se mostrará un cuadro de texto para introducir el nombre de la ruta a guardar:

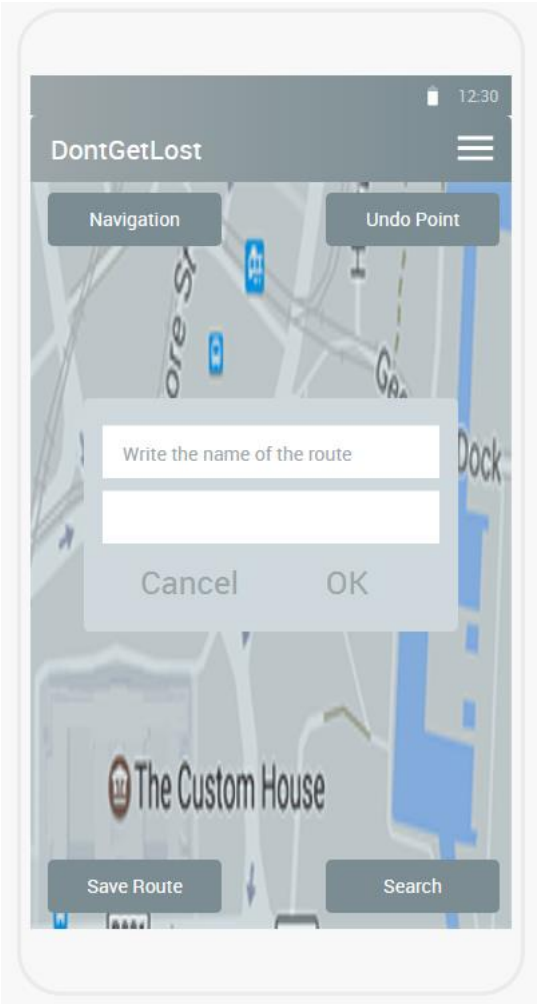

Figura 10: Diseño UI guardar ruta

Pulsando en el menú desplegable, se mostrarán tres posibles opciones: ver puntos habituales, crear ruta con puntos habituales y añadir puntos habituales.

El proceso de añadir puntos habituales es similar al de búsqueda. Si se pulsa en esta opción, se mostrará la misma interfaz de búsqueda que se explicó con anterioridad, salvo que cuando se seleccione la opción que se desea se mostrará un cuadro de texto en el que introducir el nombre del punto habitual a guardar.

La interfaz de ver puntos habituales es similar a la del menú principal.

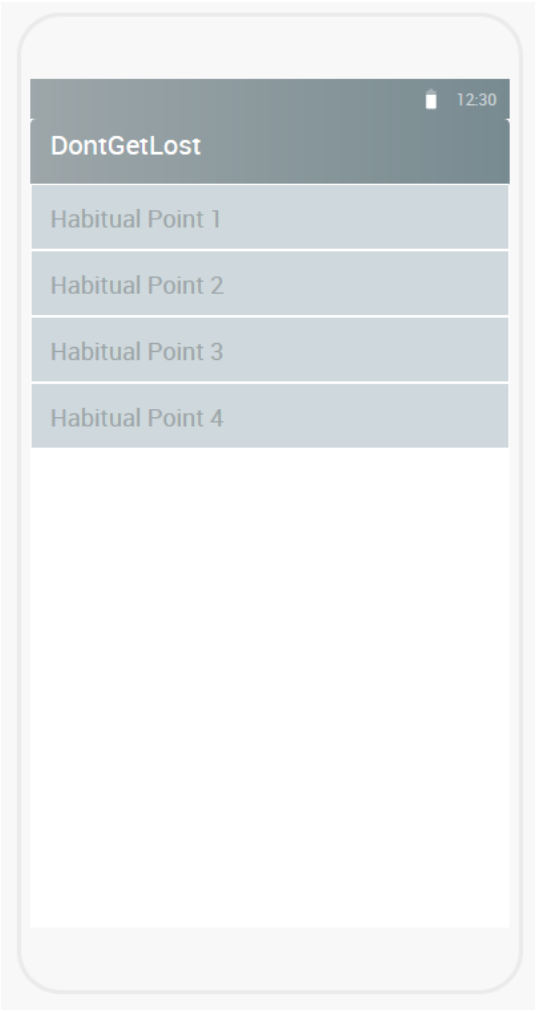

Figura 11: Diseño UI ver puntos habituales

Aquí se podrán ver los puntos habituales, en formato de lista, que se hayan guardado.

Una vez creados varios puntos habituales, se puede crear una ruta con puntos habituales.

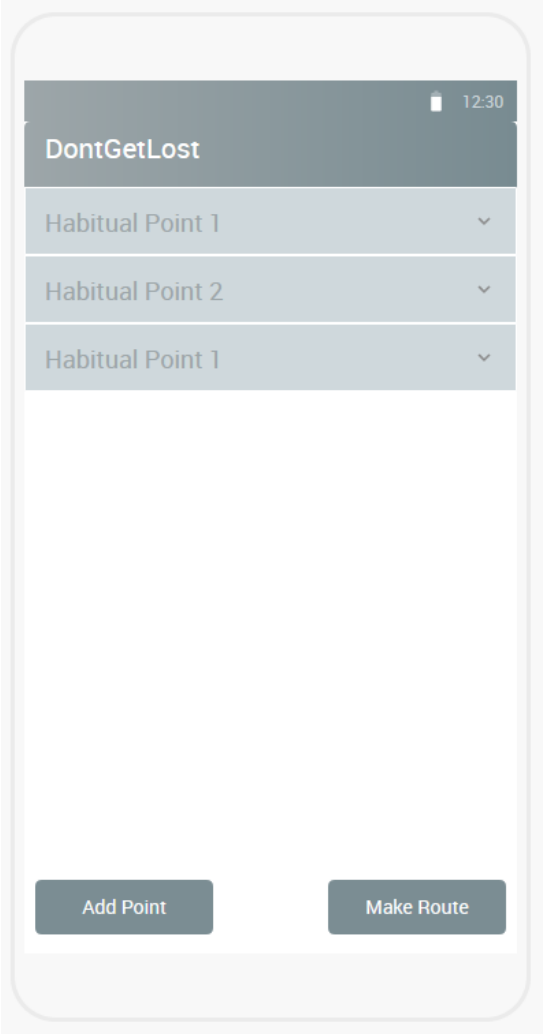

Figura 12: Diseño UI crear ruta con puntos habituales

El formato es muy sencillo, pero muy potente. Se trata de una lista de puntos que se irá modificando, escogiendo aquellos que interese para crear la ruta que se desea. Cuando se pulsa en un punto, la aplicación muestra un menú desplegable, llamado Spinner en Android, que muestra los posibles puntos que se pueden seleccionar. Las opciones que la aplicación muestra son únicamente aquellos puntos que se hayan guardado con anterioridad, por lo que la búsqueda de localizaciones previamente guardadas es mucho más sencilla y rápida.

Se tiene la posibilidad de añadir más puntos pulsando el botón de añadir punto situado en la parte inferior izquierda. Una vez se haya terminado la ruta deseada, se pulsará el botón crear ruta, tras lo cual la aplicación se encargará automáticamente de crear la ruta con los nombres de los puntos que se hayan seleccionado. Esta ruta aparecerá en el menú principal junto con las demás rutas.

4 Desarrollo

Una vez realizado el diseño de la aplicación, es hora de centrarse en el desarrollo de esta.

Para llevar a cabo esta labor, se ha utilizado Android Studio como IDE, ya que facilita enormemente la labor de desarrollo de aplicaciones Android. Debido a la complejidad de crear un sistema de Mapas y GPS desde cero, se ha utilizado un framework llamado MapBox para la instanciación de los mapas, el sistema de navegación y el sistema de búsqueda de lugares de interés.

4.1 Desarrollo de los módulos

Inicialmente, se hablará del módulo de la navegación GPS y geolocalización. Como ya he dicho en la introducción a esta sección, por tratarse de un desarrollo de unas dimensiones desproporcionadas para el trabajo en cuestión, se tomó la decisión de utilizar de base un código existente que se integrara y modificara para complacer las funciones y métodos que esta aplicación demandaba. El esqueleto que se ha utilizado en este caso es MapBox. Se trata de un sistema de geolocalización y servicio GPS que facilita el desarrollo de aplicaciones que implican un mapa y una navegación. Algunos de los módulos utilizados de este servicio son la habilidad de geolocalizar al usuario y el uso de su sistema de navegación curva a curva para guiar de un punto a otro, el cual ha sido modificado para albergar la posibilidad de navegación multipunto.

Esta labor se realiza en formato de caja negra, ya que el sistema interno de MapBox es complejo. La aplicación que se está desarrollando se encarga de tratar la información del usuario adecuadamente para que esta sea enviada de forma correcta al sistema MapBox, y recibir la respuesta que posteriormente será procesada y mostrada al usuario. Esta información es la relativa al cálculo de rutas, búsqueda de puntos de interés y navegación.

La información gráfica del mapa es extraída del sistema Mapbox, al cual se le modifican los parámetros para adecuarlo a la aplicación.

La gestión de rutas se realiza de forma local en el dispositivo. Se guarda en ese lugar la información relevante de las rutas que el usuario ha querido guardar. Este módulo también se encarga de leer dicha información y enviarla, bien al módulo de interfaz gráfica para que

se muestre en el menú principal o bien al módulo del mapa para que guarde este circuito como itinerario actual.

El módulo de interfaz gráfica se ha desarrollado utilizando las herramientas de implementación de interfaces de Android Studio, que combina funcionalidades que permiten arrastrar componentes a la interfaz y un desarrollo basado en un formato XML. Por lo que, en líneas generales, se utiliza el arrastrado de componentes para, seguidamente, realizar modificaciones en el código XML generado para adecuarlo a nuestra aplicación.

Por último, el módulo de puntos habituales gestiona de forma local los puntos que el usuario haya creado. La información se guarda en un fichero diferente al de las rutas para una mejor organización. Siguiendo la misma línea que el módulo de rutas, este módulo lee la información de los puntos y los envía a la interfaz para que sean vistos en un formato de lista por el usuario. Este módulo también es el encargado de mostrar los puntos posibles a la hora de crear una ruta con los puntos habituales creados, tras lo cual se envía toda la información al módulo de gestión de rutas para que este genere una ruta adecuada con los puntos escogidos.

4.2 Desarrollo de los Componentes

Partiendo de un diseño sencillo de lo que se quería conseguir, se ha llegado a la siguiente implementación, en cuanto al menú principal se refiere:

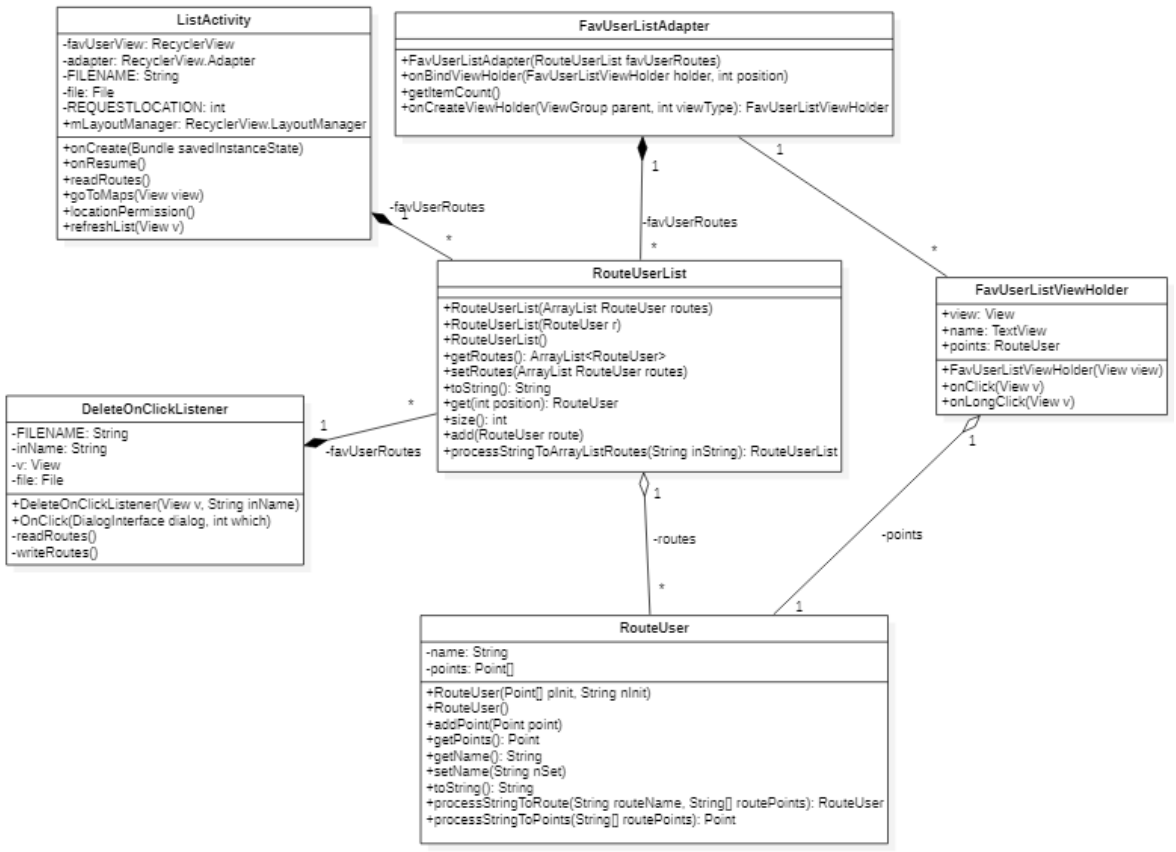

Figura 13: Implementación de la aplicación

Este es el punto de entrada de la aplicación. La actividad principal es ListActivity que representa el Menú Principal. Toda actividad Android comienza por OnCreate. En este método se realizan las operaciones de inicialización. Se comienza por comprobar si la aplicación tiene permisos de localización y, si no es así, se piden con locationPermissions(). Tras esto, se procede a leer el fichero del usuario, cuyo nombre está almacenado en FILENAME, con el método readRoutes(), por si este tuviera rutas guardadas. Este método guarda en la variable favUserRoutes las rutas parseadas del fichero interno del usuario para que sean mostradas en pantalla. Para el muestreo de estas rutas primero se asocia la variable favUserView, que contiene el formato con el cual se mostrará la información, con su homónimo en la definición XML de la interfaz. A continuación, con la variable adapter se crea un adaptador para los datos a mostrar, que sería como un contenedor de datos, y se

asocia con la variable anterior. Estas dos operaciones indican al sistema Android cómo y qué va a mostrar por pantalla. Ya que nuestros datos no son estándar, es necesario crear dos clases nuevas, FavUserViewHolder y FavUserListAdapter, correspondientes al formato de la interfaz y al contenedor de datos. En la clase de FavUserViewHolder se indican las operaciones relativas a la interfaz, el pulsado y el pulsado largo, denominadas OnClick() y OnLongClick(). Ambos son métodos de escucha de eventos, los cuales esperan a que el usuario realice una acción en concreto para activarse. Estas operaciones desencadenan dos corrientes muy concretas.

La primera se refiere a pinchar en una ruta que se haya mostrado en el Menú Principal. Tras esta operación, se capta cuál ha sido la ruta que se ha pulsado, se guarda en un formato especial de Android llamado Bundle y se inicia la actividad del Mapa. Este formato especial contiene la información de la ruta que se ha seleccionado, con lo que la actividad de Mapa, tras analizar esta información, genera la ruta correcta.

La segunda operación, pulsado largo, es la relativa a la eliminación de una ruta. Este escuchador recibe la información y extrae cuál ha sido la ruta seleccionada para la eliminación. Ya que esta operación era un poco más compleja, se decidió crear una clase especial que la albergase, DeleteOnClickListener. Esta clase recibe únicamente cual ha sido la ruta seleccionada, generando el cuadro de dialogo de confirmación. Si el usuario confirma, se ejecutan varias operaciones. Primero se lee el fichero que contiene todas las rutas, readRoutes(). Tras lo cual, se elimina la ruta seleccionada. Una vez eliminada, se escribe la nueva lista de circuitos en el fichero, completando así el eliminado.

Se tuvieron muchos problemas a la hora de actualizar automáticamente la interfaz, por lo que se optó por introducir un botón destinado a ello. Al pulsar este botón se realiza la operación refreshList() de la actividad principal ListActivity, que actualiza la lista de rutas y la interfaz gráfica.

Por último, la operación de iniciar el mapa sin cargar ninguna ruta se realiza con el botón destinado a ello. Este botón activará el método goToMaps() que únicamente inicia la actividad mapa de forma estándar, sin información adicional.

Ya que las rutas y puntos no son un tipo de datos estándar, se han tenido que crear dos clases nuevas para definir estos tipos de datos.

Primero, RouteUser, el cual contiene un nombre y una serie de puntos. Con sus operaciones estándar de crear la ruta, añadir puntos, etcétera. El método habitual para su creación es pasando como argumentos el nombre y la serie de puntos. Es notable mencionar dos métodos muy importantes, processStringToRoute() y processStringToPoints(). El primero es el encargado de manejar las operaciones de creación de rutas a partir de Strings cuando se lee el fichero de la memoria interna del dispositivo. El segundo es el encargado de parsear la información de la ruta que se le ha pasado al mapa cuando éste se inicializa con una ruta cargada. El formato especifico de cómo se debe parsear una ruta a String se define en el método toString().

El otro tipo de datos creado es RouteUserList, que sirve para gestionar una lista de RouteUser. Sus creadores habituales son el paso de un solo RouteUser o el paso de una lista de RouteUser. El método processStringToArrayListRouteUsers() es el encargado de parsear por completo el fichero de rutas. Este método se apoya en el método de procesado de la clase RouteUser para el parseo individual de cada ruta.

Siguiendo con el desarrollo, esta es la implementación de la actividad del mapa:

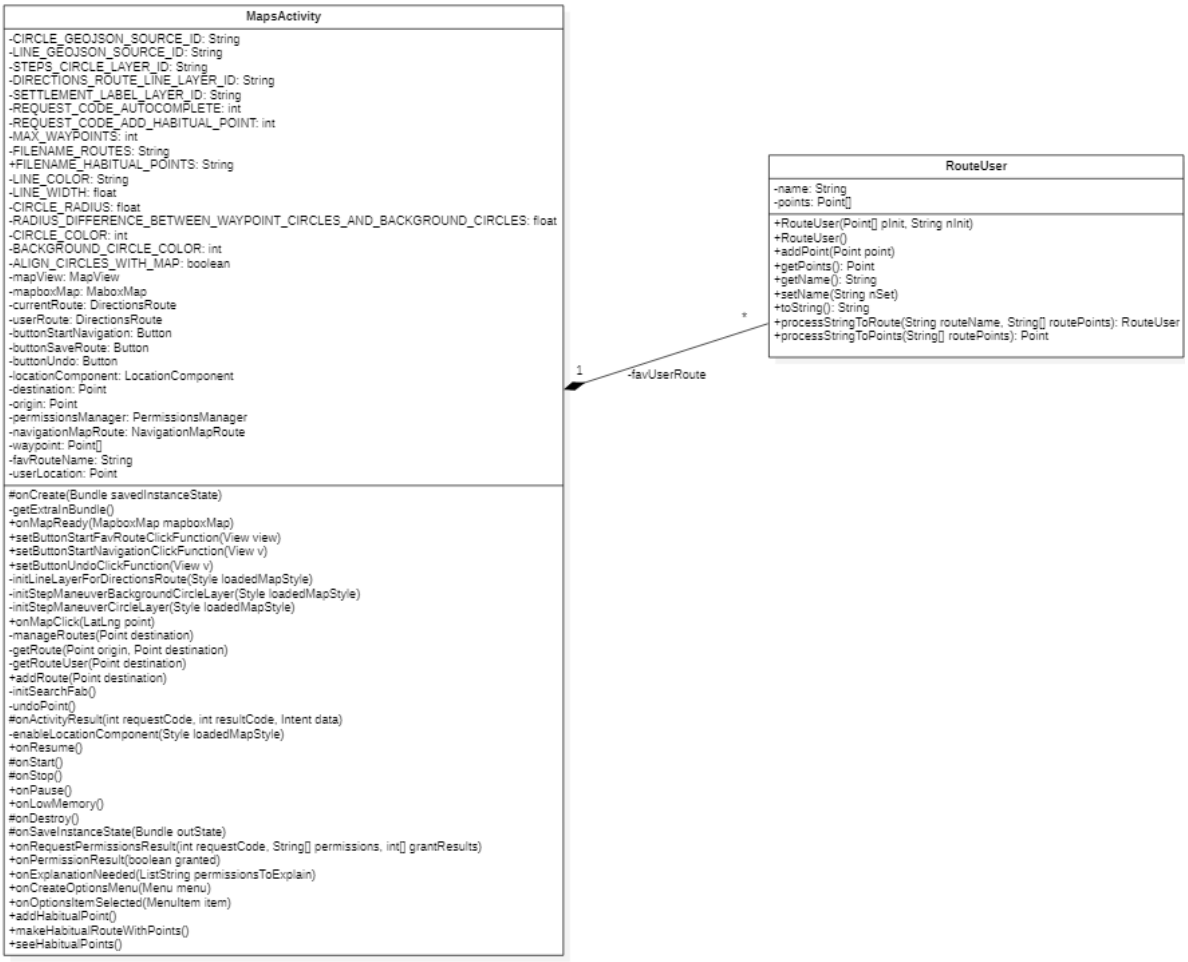

Figura 14: Implementación de la actividad mapa

La actividad del mapa mezcla funcionalidad introducida del framework de Mapbox con aquella que se ha añadido con la finalidad de ampliar la funcionalidad de la aplicación.

Existen numerosas constantes destinadas a la definición de la parte gráfica del mapa. Como en la actividad anterior, en el método OnCreate() se asocia la variable del formato con la definición gráfica XML, en este caso mapView.

Tras esto, se invoca OnMapReady() que inicializa el contenedor mapboxMap, que es el encargado de albergar tanto los estilos del mapa como la funcionalidad que en él reside. Aquí se invocan los métodos de estilos: initLineLayerForDirectionsRoute(), que indica el estilo de la línea de la ruta; initStepManeuverCircleLayer(), que indica el estilo de los puntos en el mapa, e initStepManeuverBackgroundCircleLayer(), que indica el fondo de los puntos en el mapa. También se inicializan los componentes de localización enableLocationComponent() y la función de búsqueda initSearchFab(). A su vez, se inicializan los botones de las

diferentes funcionalidades: buttonStartNavigation, que inicia la navegación, buttonSaveRoute, que guarda la ruta y buttonUndo, que elimina el último punto. Cada uno de estos botones invoca a sus respectivos métodos: setButtonStartnavigationClickFunction(), para el inicio de la navegación; setButtonSaveRouteClickFunction, para el guardado de ruta, y setButtonUndoClickFunction, para deshacer el último punto introducido.

Cuando el mapa ya se ha inicializado, antes de mostrar al usuario la información precisa, se comprueba si se ha cargado alguna información adicional cuando se inició la actividad. Si es así se invoca el método getExtrasInBundle(), que extrae esa información y la almacena en la ruta actual variable denominada currentRoute. Tras esta operación, se invoca el sistema de generación de rutas y se muestra la ruta cargada.

Si no se ha indicado ninguna ruta, se muestra el mapa con la localización del usuario. Cuando este pulse en el mapa, se invoca el método OnMapClick(), que extrae las coordenadas del punto pulsado, y llaman al método manageRoutes(). Este método es el punto de entrada del generador de rutas. Según las condiciones en las que se encuentre la ruta, se llamará al método getRoute(), con el origen y destino si es el primer punto, o a addRoute(), únicamente con el destino si es un punto adicional. Estos métodos, en esencia, realizan la misma operación, generar una ruta entre los puntos origen y destino. La peculiaridad del método addRoute() es que tiene en cuenta los llamados waypoints, puntos intermedios de ruta, pudiendo así crear una ruta multipunto. Una vez generada la ruta, se muestra en el mapa.

La operación de búsqueda se inicia pulsando el botón de la lupa. Si el usuario pulsa este botón, se ejecuta una actividad auxiliar que muestra el cuadro de texto y las opciones que se pueden seleccionar. Tras introducir lo que se busca y seleccionar algún resultado, este se pasa al método OnActivityResult(), con un código específico a esta operación, REQUEST_CODE_AUTOCOMPLETE. Este método extrae las coordenadas del lugar buscado, con un sistema proporcionado por el framework de Mapbox, y acerca la cámara a dicho punto añadiéndolo a la ruta con el mismo método que se utilizó con anterioridad, manageRoutes().

Como se ha explicado en el diseño, el botón de navegación comienza la actividad de navegación que está proporcionada por el framework de Mapbox. Dicho esto, antes de invocar a la navegación, la aplicación utiliza la localización del usuario como origen y llama

al método getRouteUser, que genera una ruta desde la localización de usuario hasta el destino pasando por todos los puntos intermedios, que en este caso como mínimo será el origen de la ruta formada con anterioridad. Esta ruta se denomina userRoute, ya que es de interés mantener por separado la ruta que el usuario navega con la que ha creado el usuario.

El botón de deshacer punto invoca únicamente al método undoPoint() cuando se haya agregado algún punto al mapa, de lo contrario mostrará un mensaje de error indicando que no se puede deshacer ningún punto. Este método deshace el último punto agregado e invoca de nuevo al generador/manejador de rutas, manageRoutes(), para generar la nueva ruta y mostrársela al usuario.

El botón de guardar ruta mostrará un cuadro de diálogo únicamente si hay una ruta en el mapa, de lo contrario mostrará un mensaje de error en pantalla. Tras indicar el nombre de la ruta, la aplicación analiza las rutas ya creadas y, si existe una ruta con el mismo nombre, indicará al usuario el error; de lo contrario, guardará la ruta en el fichero interno del dispositivo, parseando la ruta como String.

Por último, las diferentes opciones que se muestran en el menú desplegable de la parte superior derecha de la interfaz gráfica conllevan una llamada a los métodos correspondientes.

Estos métodos son: addHabitualPoint(), makeHabitualRouteWithPoints() y seeHabitualPoints(). Cada uno de estos métodos inicia la actividad relativa a su funcionalidad. Los métodos makehabitualRouteWithPoints() y seeHabitualPoints() inician respectivamente ListMakehabitualRoute y ListHabitualPointsActivity. Mientras que addhabitualPoint() inicia una actividad auxiliar que se comporta de forma similar a la de búsqueda de punto de interés. En esta actividad auxiliar se muestra un cuadro de texto y según se escriba se muestran opciones posibles. Al seleccionar una, la actividad se cierra y se inicia OnActivityResult con el código correspondiente a esta operación, REQUEST_CODE_ADD_HABITUAL_POINT. Se extrae la información del punto escogido y se muestra al usuario un cuadro de dialogo para introducir el nombre del punto.

Tras el desarrollo del mapa, se puede pasar a la implementación de la lista de puntos habituales y su gestión.

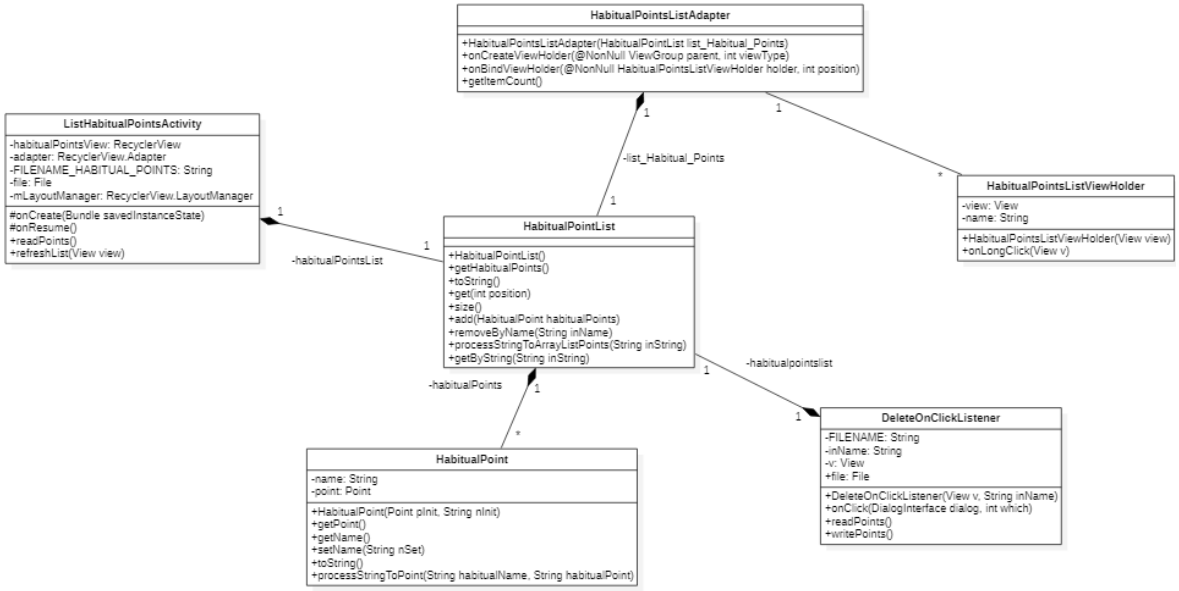

Figura 15: Implementación de la lista de puntos habituales

El punto de entrada en esta actividad es ListHabitualPointsActivity. En concepto es muy parecida a ListActivity, ya que actúa de interfaz gráfica y de gestión de puntos habituales, en vez de rutas. Su interfaz es parecida, tratándose de un listado de los puntos habituales que el usuario tenga guardados.

Se inicia en OnCreate() y se comienza por leer el fichero, con el nombre indicado en FILENAME, donde se albergan los puntos habituales, inicializando la variable habitualPointsList utilizando el método readPoints(). También se asocia el formato de la interfaz gráfica con su homónimo en la definición de la interfaz en XML, habitualPointsView. Tras esto, se inicializa el adaptador, adapter, con los datos de habitualPointsList y se asocia al View. Esto indica al sistema lo explicado con anterioridad, el qué y el cómo mostrar los datos.

Se crearon igualmente las clases pertinentes HabitualPointsListAdapter y HabitualPointsListViewHolder, correspondientes al formato del contenedor e interfaz de los datos. En el componente de la interfaz, el ViewHolder, se definió el pulsado largo. Esta operación permite eliminar un punto habitual que ya no se utilice. Se define en dos pasos. Primero, un escuchador de eventos, que espera a que el usuario realice la acción pertinente, en este caso OnLongClick(). Seguido de una clase aparte, creada para esta operación, DeleteOnClickListener, explicada con anterioridad en ListActivity, que realiza la función de

leer, comprobar y eliminar del fichero de la memoria interna del dispositivo, salvo que en esta ocasión lo realiza sobre el fichero de puntos habituales. Esta operación se apoya sobre los métodos de readPoints() y writePoints().

Se crearon también los tipos de datos no estándar HabitualPoint y HabitualPointList.

HabitualPoint define un punto habitual en la aplicación. Se define con un nombre y un punto. En su constructor habitual se utilizan como argumentos de entrada estos dos datos, nombre y punto. Es notable mencionar la importancia del método processStringToPoint(), que, con la similitud de lo explicado con anterioridad, realiza una conversión de String a Punto en la lectura del fichero de guardado. Este formato está definido en el método toString(), que se invoca cuando se convierte un HabitualPoint a String para su guardado.

Por último, HabitualPointList que se trata de una lista de HabitualPoint. Contiene un método de eliminación por nombre removeByName() que busca en la lista un punto con el mismo nombre y lo elimina. Esta operación se utiliza en la eliminación de puntos habituales. El método encargado de realizar la conversión de String, desde el fichero, al formato de tipo de datos HabitualPointList es processStringToArrayListPoints(), que parsea según un formato definido en toString a la hora de guardado.

Por último, la actividad ListMakeHabitualRoute que se encargará de crear rutas con estos puntos habituales.

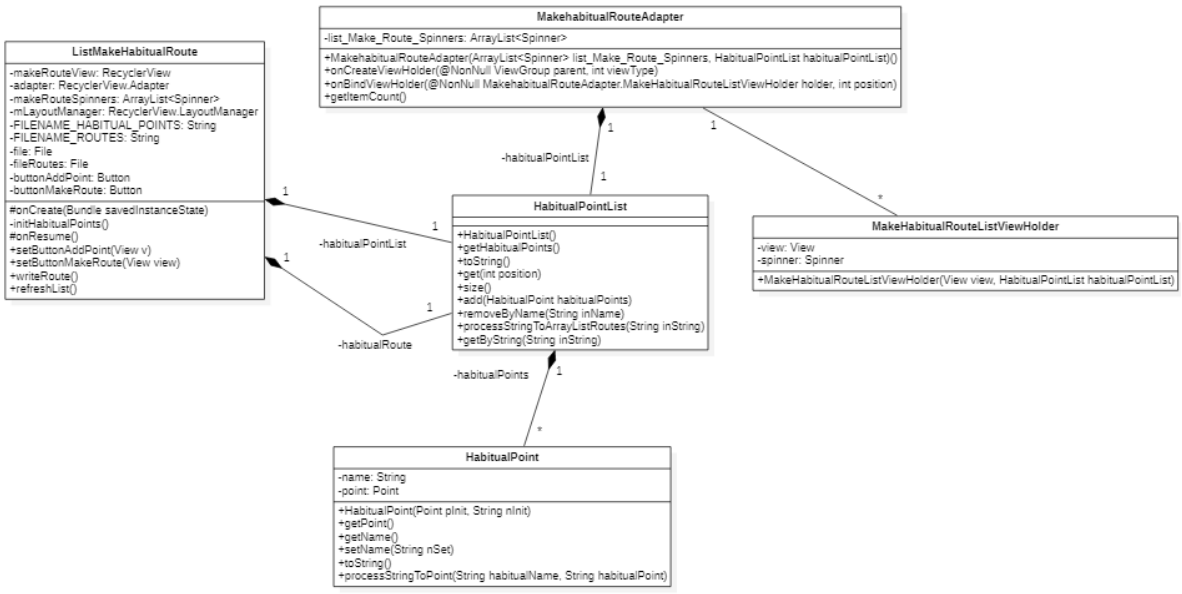

Figura 16: Implementación de la creación de rutas con puntos habituales

El punto de entrada de esta actividad es ListMakeHabitualPoints. Se trata de una lista de puntos habituales que formarán eventualmente una ruta habitual.

Como toda actividad, Android comienza en onCreate(). En primer lugar, se obtienen los ficheros que contienen tanto los puntos habituales ya creados como las rutas ya creadas, ya que será de utilidad para leer y guardar en los respectivos ficheros. A continuación, se asocian los componentes de interfaz gráfica, MakeHabitualRouteViewHolder, a los homónimos en la implementación XML y se asocia el adaptador a este, el cual albergará los datos, MakeHabitualRouteAdapter. El contenedor de datos que poblará estos componentes se inicializa en initHabitualPoints(), que lee el fichero de puntos habituales y extrae dichos puntos juntos con sus nombres. El formato que se muestra es un Spinner, un menú desplegable de opciones acotado a los puntos habituales ya creados. Así, de una manera sencilla, se puede escoger el punto deseado. Existen dos botones, buttonAddPoint y buttonMakeRoute, con sus respectivos métodos setButtonAddPoint() y setButtonMakeRoute(). El primero añade un punto a la lista de puntos de la ruta, y esto genera un nuevo Spinner. El segundo obtiene todos los puntos seleccionados, crea una ruta con ellos y la escribe en el fichero de rutas del usuario en la memoria interna del dispositivo.

4.3 Desarrollo de la Interfaz

En este apartado se verá la implementación final del diseño que se realizó previamente. Ya que se hizo esa labor, se intenta mantener fidelidad al diseño. Al igual que el desarrollo de los componentes, para la interfaz se ha utilizado también Android Studio, que facilita el trabajo de creación de GUI.

En cuanto al menú principal, tenemos esta implementación:

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".activities.ListActivity">

    <Button
        android:id="@+id/goToMapsButton"
        android:layout_width="28dp"
        android:layout_height="28dp"
        android:layout_marginEnd="32dp"
        android:layout_marginBottom="16dp"
        android:background="@drawable/ic_maps"
        android:gravity="bottom|end"
        android:onClick="goToMaps"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />

    <Button
        android:id="@+id/refreshButton"
        android:layout_width="28dp"
        android:layout_height="28dp"
        android:layout_marginStart="32dp"
        android:alpha="255"
        android:background="@android:drawable/ic_menu_rotate"
        android:gravity="bottom|start"
        android:onClick="refreshList"
        app:layout_constraintBottom_toBottomOf="@+id/goToMapsButton"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="@+id/goToMapsButton" />

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/favUserListView"
        android:layout_width="0dp"
        android:layout_height="0dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Figura 17: Implementación XML del menú principal

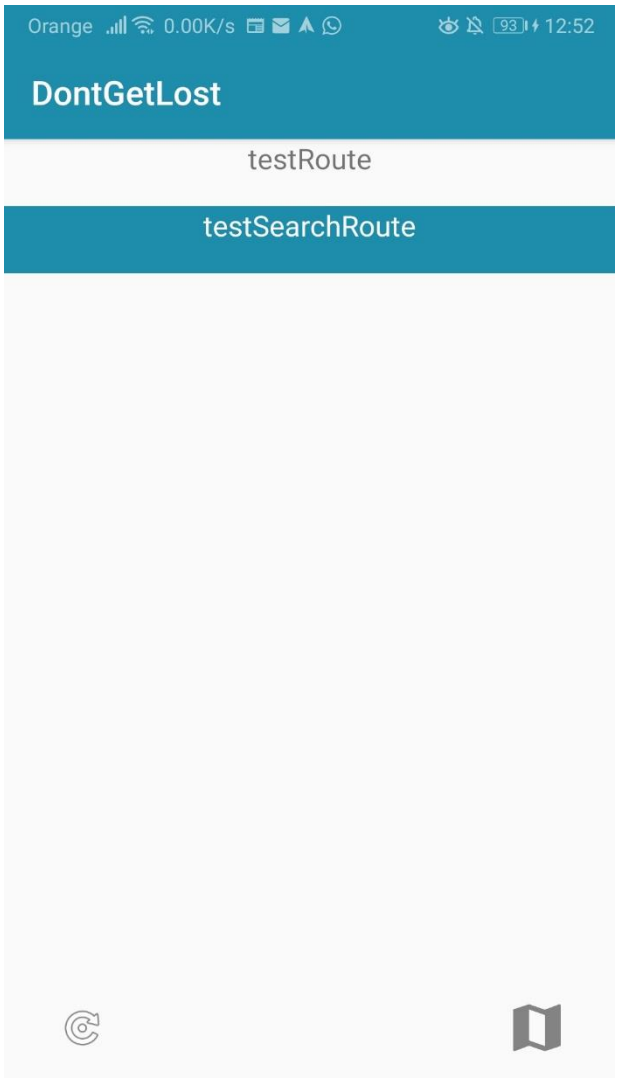

Figura 18: Implementación del menú principal

Como se puede observar, se ha intentado mantener la sencillez sin prescindir de funcionalidad y estética. Se tiene un título, el cual es constante en todas las interfaces, con el nombre de la aplicación, en un color azul, seguido de la lista de rutas que el usuario haya guardado. Para mayor comodidad y legibilidad, se ha optado por alternar colores entre rutas aumentando así también la estética de la aplicación. En la parte inferior tenemos los dos botones de acción. A la derecha, el botón de iniciar el mapa. Este botón llevará al usuario al mapa sin cargar ninguna ruta. A la izquierda, el botón de refrescar, que tendrá la función de refrescar la interfaz una vez se haya eliminado una ruta.

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:mapbox="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <com.mapbox.mapboxsdk.maps.MapView
        android:id="@+id/mapView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        mapbox:mapbox_cameraTargetLat="38.9098"
        mapbox:mapbox_cameraTargetLng="-77.0295"
        mapbox:mapbox_cameraZoom="12" >

        <com.google.android.material.floatingactionbutton.FloatingActionButton
            android:id="@+id/fav_location_search"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="end|bottom"
            android:layout_margin="16dp"
            android:backgroundTint="@color/mapbox_blue"
            android:src="@drawable/ic_search"
            android:tint="@color/mapboxWhite"
            mapbox:backgroundTint="@color/mapboxBlue"
            mapbox:rippleColor="@color/mapboxWhite" />

        <Button
            android:id="@+id/startButton"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="start|top"
            android:layout_margin="16dp"
            android:backgroundTint="@color/mapbox_blue"
            android:enabled="true"
            android:onClick="setButtonStartNavigationClickFunction"
            android:text="Start navigation"
            android:textColor="@color/mapboxWhite"
            android:visibility="visible" />

        <Button
            android:id="@+id/saveRoute"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="start|bottom"
            android:layout_margin="16dp"
            android:backgroundTint="@color/mapbox_blue"
            android:enabled="true"
            android:onClick="setButtonSaveRouteClickFunction"
            android:text="Save Route"
            android:textColor="@color/mapboxWhite"
            android:visibility="visible" />

        <Button
            android:id="@+id/undo"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="top|end"
            android:layout_margin="16dp"
            android:backgroundTint="@color/mapbox_blue"
            android:onClick="setButtonUndoClickFunction"
            android:enabled="true"
            android:text="Undo Last Point"
            android:textColor="@color/mapboxWhite"
            android:visibility="visible" />

    </com.mapbox.mapboxsdk.maps.MapView>

</androidx.constraintlayout.widget.ConstraintLayout>
```

Figura 19: Implementación XML interfaz mapa

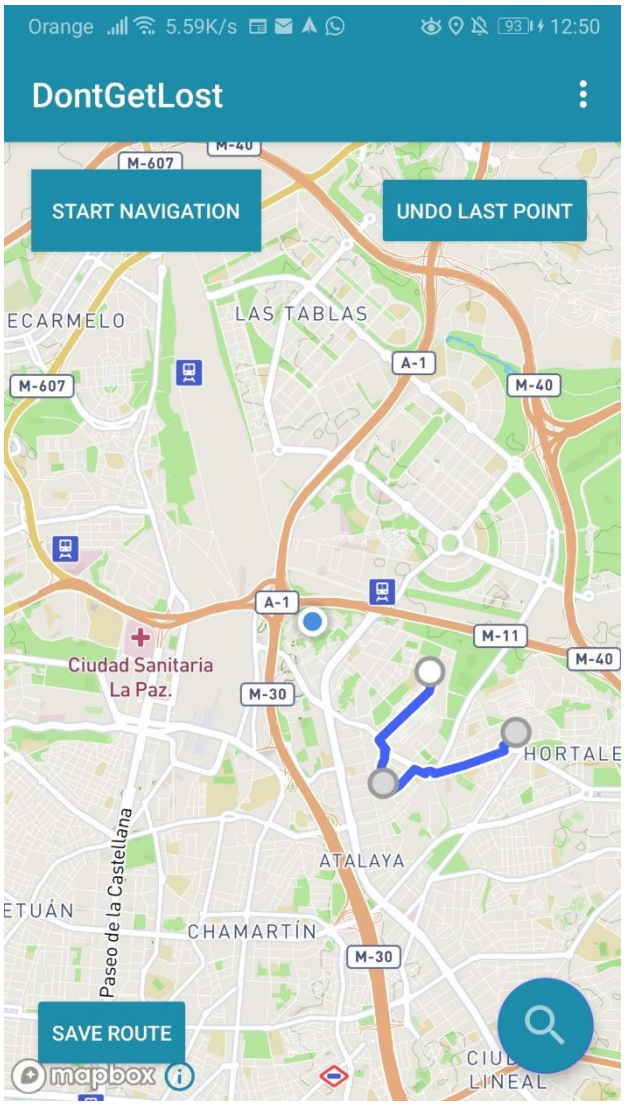

Figura 20: Implementación UI mapa con ruta

Una vez iniciemos la actividad del mapa, se podrá ver un mapa con información de interés, así como una clara interpretación de la ruta que estemos diseñando. Se mostrará con un punto nuestra posición, tras la cual podremos comenzar a diseñar la ruta. También obtendremos la información del circuito que se esté creando con unas indicaciones gráficas, en forma de líneas y puntos interconectados.

Existen cuatro botones bien diferenciados. Primero, arriba a la izquierda, el botón de navegar, que iniciará la navegación. A la derecha, el botón de deshacer punto, que eliminará el último punto introducido. A continuación, en la parte inferior izquierda, el botón de guardar ruta, que guardará la ruta. A la derecha, el botón de buscar, que iniciará la actividad auxiliar de búsqueda de punto de interés. Como se puede ver, se ha conservado una amplia

parte de la interfaz para el mapa, para maximizar la información que el usuario recibe. Los botones tienen un color similar al tema de la aplicación, por lo que la estética se conserva, y se sitúan en las esquinas de la pantalla, para minimizar su impacto pero tener máxima accesibilidad.

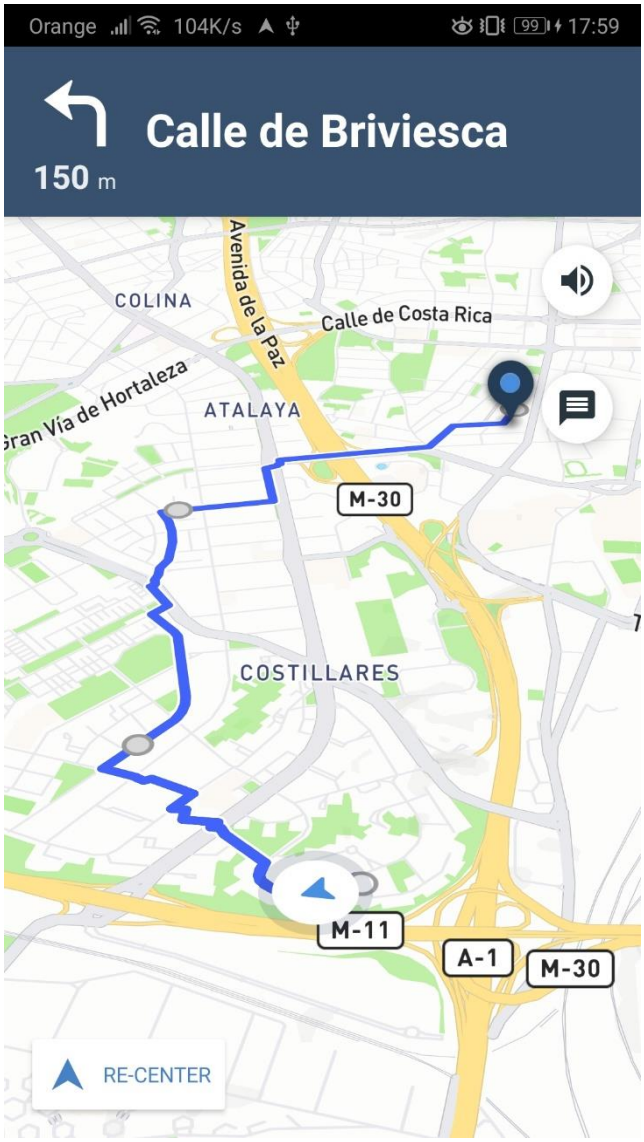

Figura 21: Implementación UI navegación

La interfaz de la navegación, como se puede ver, es sencilla y precisa de los componentes necesarios. Se mostrará al usuario con forma de flecha azul encima de la línea de la ruta que tendrá que seguir con las indicaciones que la aplicación le ofrece. En la parte inferior se mostrará información -tiempo, distancia, etcétera- relevante a la ruta.

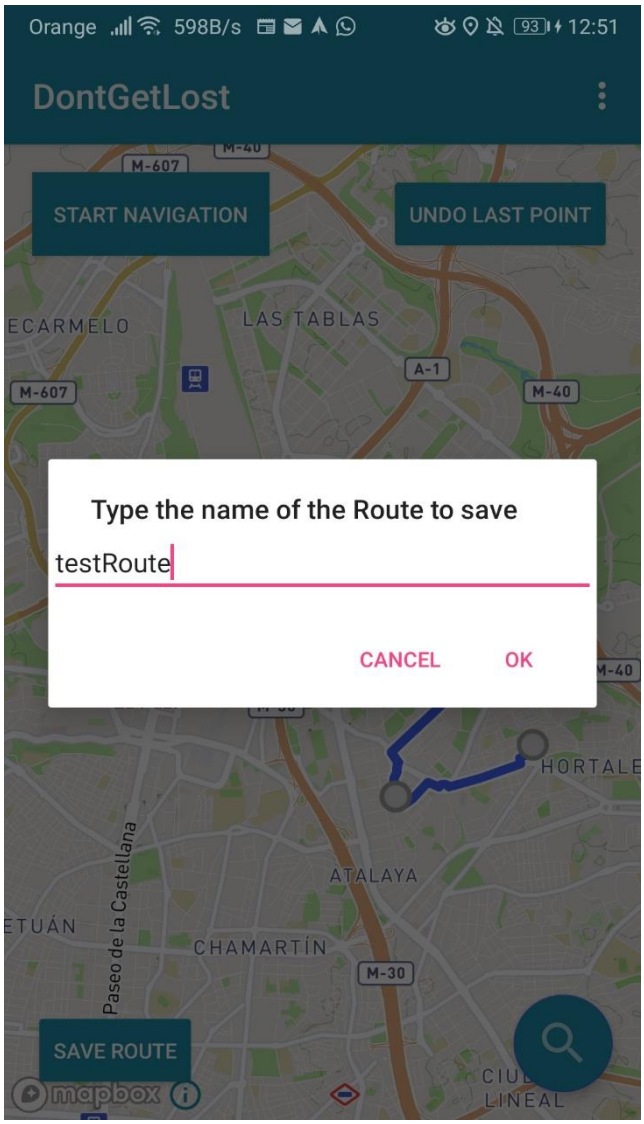

Figura 22: Implementación del guardado de ruta

Aquí se puede ver el cuadro de diálogo que se mostrará una vez pulsemos el botón de guardar ruta.

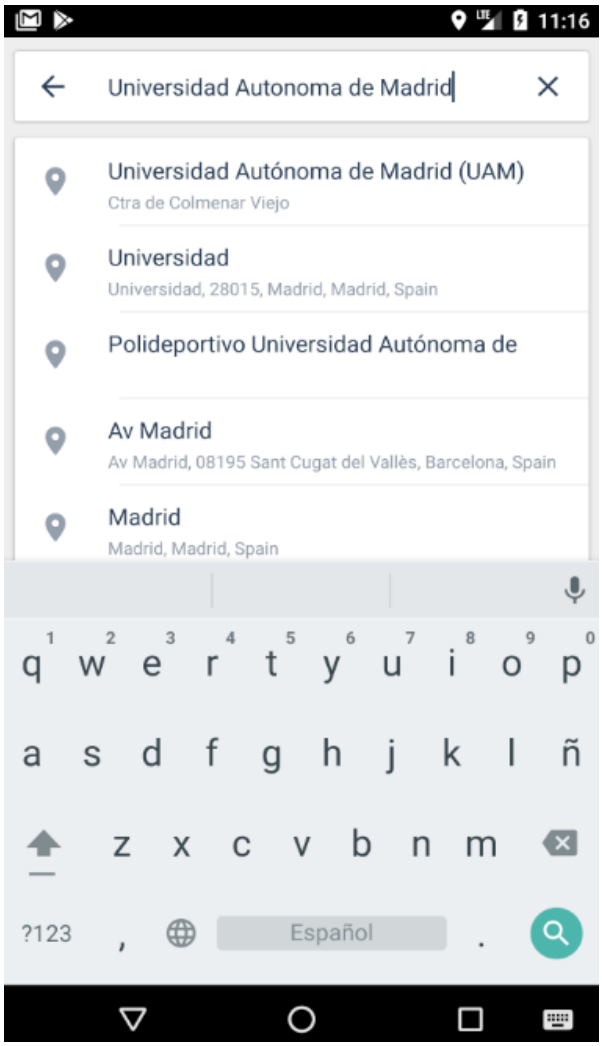

Figura 23: Implementación búsqueda

Si, en cambio, se pulsa el botón de búsqueda, se lanzará una actividad auxiliar que cuenta con un cuadro de texto en la parte superior y sugerencias del sistema de búsqueda en la parte inferior. En este caso, a medida que se vaya introduciendo texto, la aplicación mostrará sugerencias que se adecuen a lo introducido. El usuario podrá seleccionar el resultado que sea, el que busca, activando la acción de añadir dicho punto a la ruta que se esté formando.

Si se selecciona el desplegable de la interfaz de mapa, se obtendrá un menú despegable con tres opciones: añadir punto habitual, crear ruta con puntos habituales y ver puntos habituales.

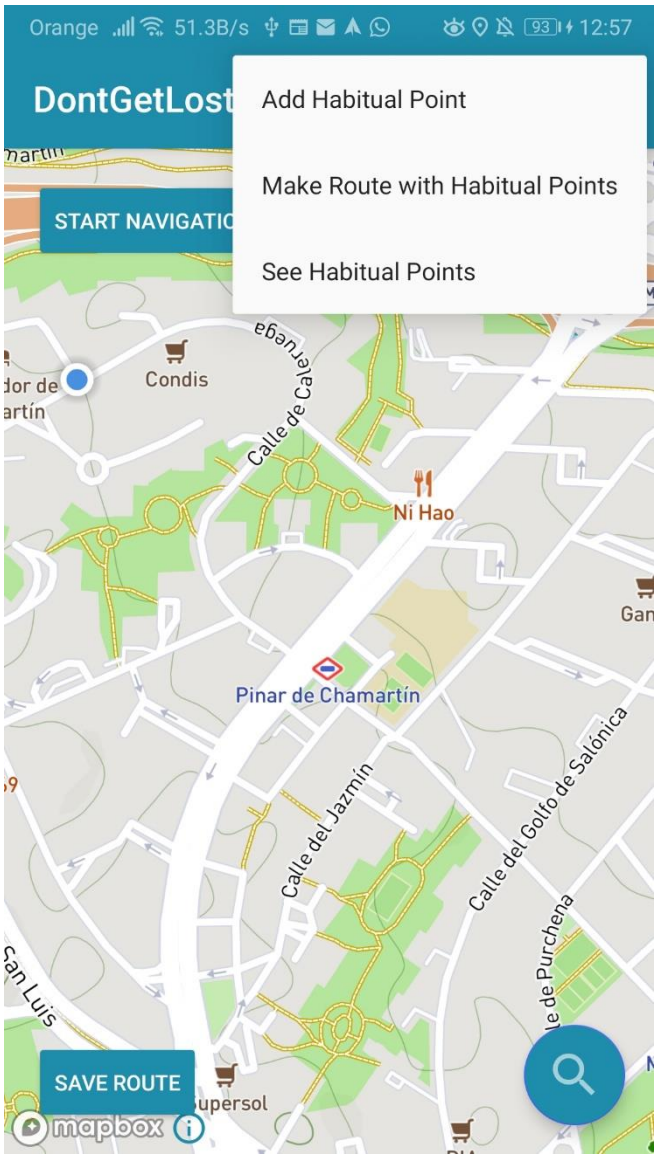

Figura 24: Interfaz acceso puntos habituales

Primeramente, la de creación de puntos habituales, que utiliza el mismo proceso de búsqueda que cualquier otro punto. Una vez escogido un punto, saltará de nuevo un cuadro de diálogo para darle nombre al punto.

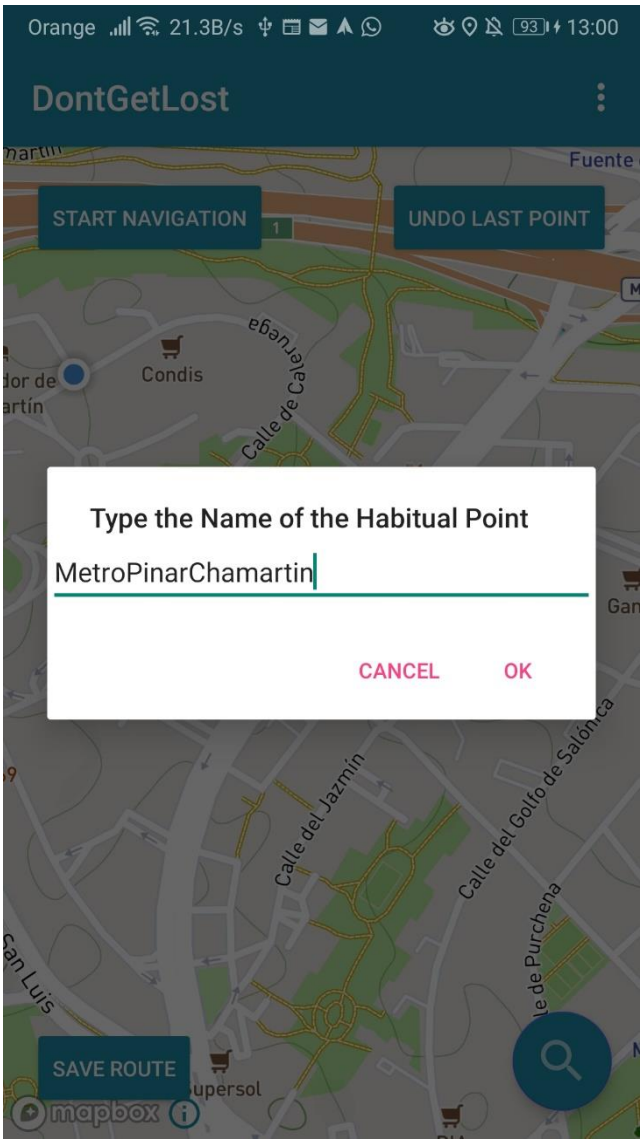

Figura 25: Implementación nombre a punto habitual

Tras esta operación, se podrá ir a visualizar los puntos creados.

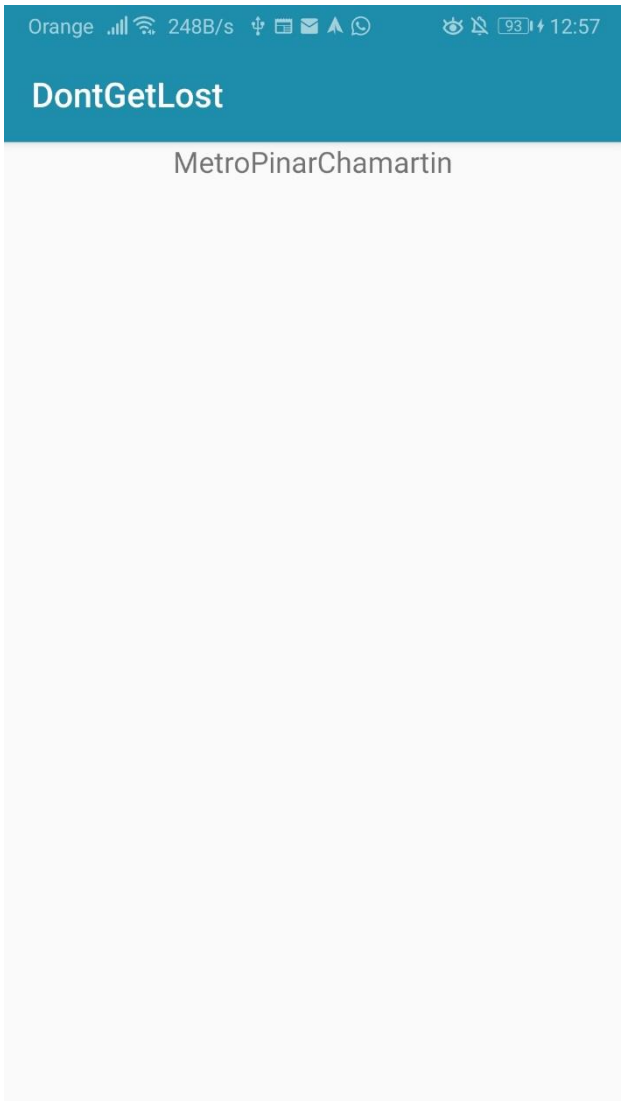

Figura 26: Implementación visualizar puntos habituales

Se puede observar un patrón similar al del Menú Principal. Se trata de una lista de puntos habituales, que en este caso también se alternará de colores para una mejora estética y una mejor legibilidad. Ya que en esta interfaz la única funcionalidad viene dada con el pulsado largo, no existen más botones.

Y, por último, crear una ruta con puntos habituales.

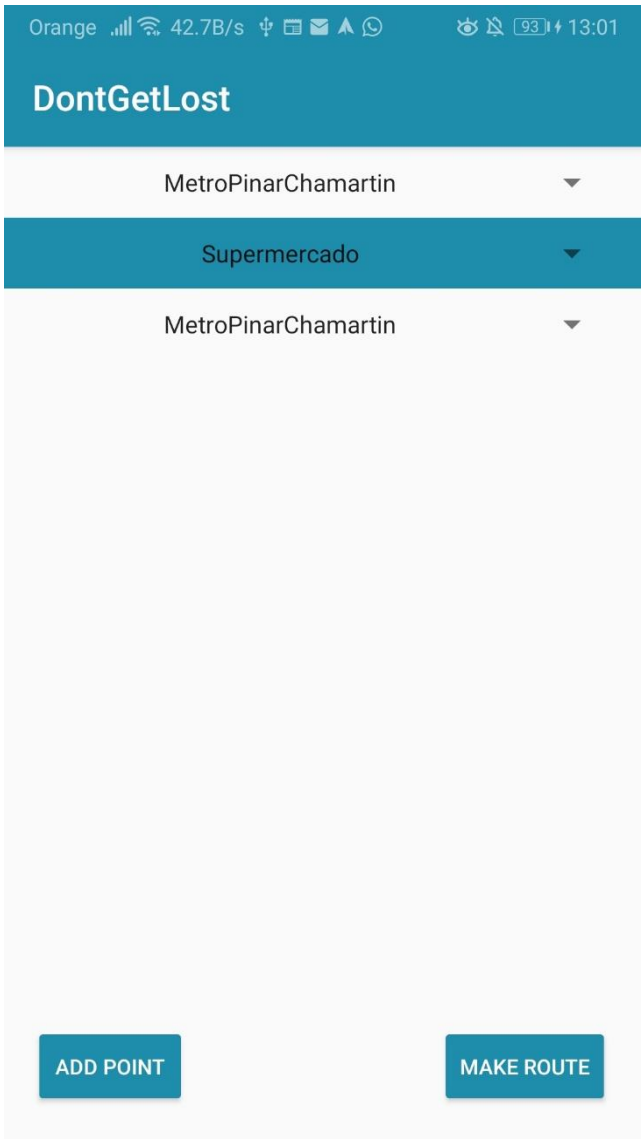

Figura 27: Implementación ruta con puntos habituales

Aquí se puede observar cómo se ha organizado la introducción de puntos habituales para la creación de rutas habituales. Se conserva una disposición parecida al del resto de interfaces, una lista de puntos con el nombre con alternancia de colores. En este caso, los componentes que forman la lista se denominan Spinners, y son una clase de contenedores que albergan un menú desplegable si se pulsa en ellos. Tras su pulsación, se mostrarán las opciones disponibles al usuario.

En la parte inferior tenemos dos botones azules, conservando la temática de la aplicación. Situado en la izquierda, el botón de añadir punto, que añadirá un punto a la lista, que se transforma en añadir un Spinner en la interfaz. Y, situado a la derecha, el botón de crear ruta, que generará la ruta con los puntos seleccionados.

5 Integración, pruebas y resultados

Para la realización de las pruebas, se utilizaron dos métodos.

Inicialmente, se utilizó un móvil virtual creado dentro del IDE Android Studio, con el que se optó en el proceso de desarrollo por la comodidad e integración que se obtiene.

Aunque a la hora de ejecutar las pruebas propiamente dichas y las incluidas en esta memoria se tomó la decisión de utilizar un móvil real, el mío personal, para dicha tarea.

Al tratarse de una aplicación GPS, la primera prueba que se lanzó fue la comprobación de que pedía correctamente el permiso de localización al usuario y, tras lo cual, encontraba a dicho usuario en el mapa.

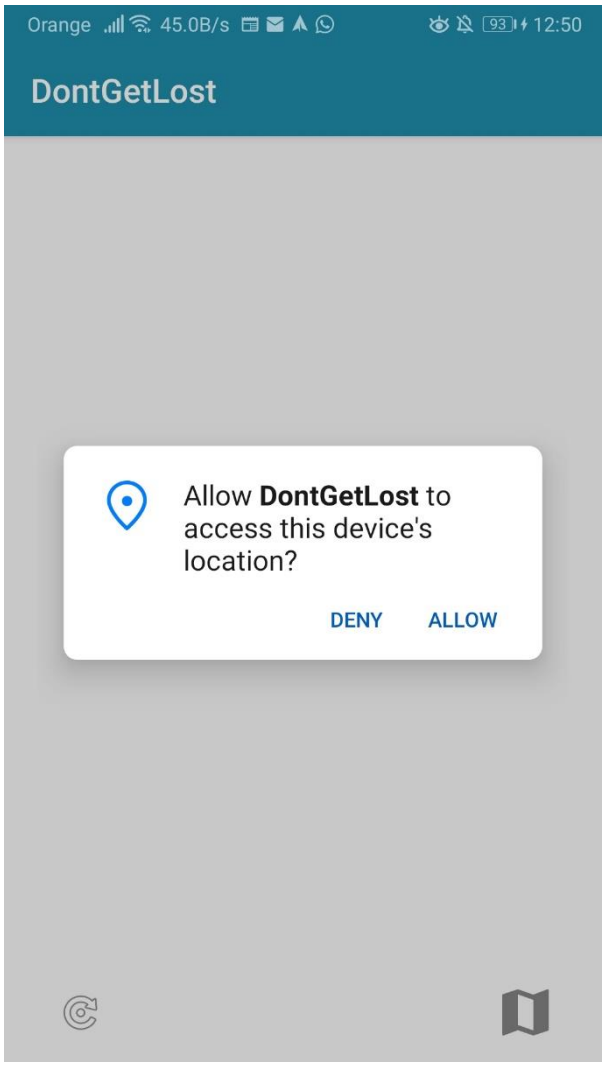

Figura 28: Petición de localización al usuario

En esta imagen vemos cómo, tras iniciar la aplicación por primera vez, esta detecta que el permiso no ha sido concedido, con lo cual trata de pedirlo.

Tras aceptarlo, nos encontramos en el menú principal, que se encuentra vacío ya que no hay rutas guardadas.

Si pulsamos el botón del mapa, nos trasladaremos al mapa con una indicación de nuestra localización en él.

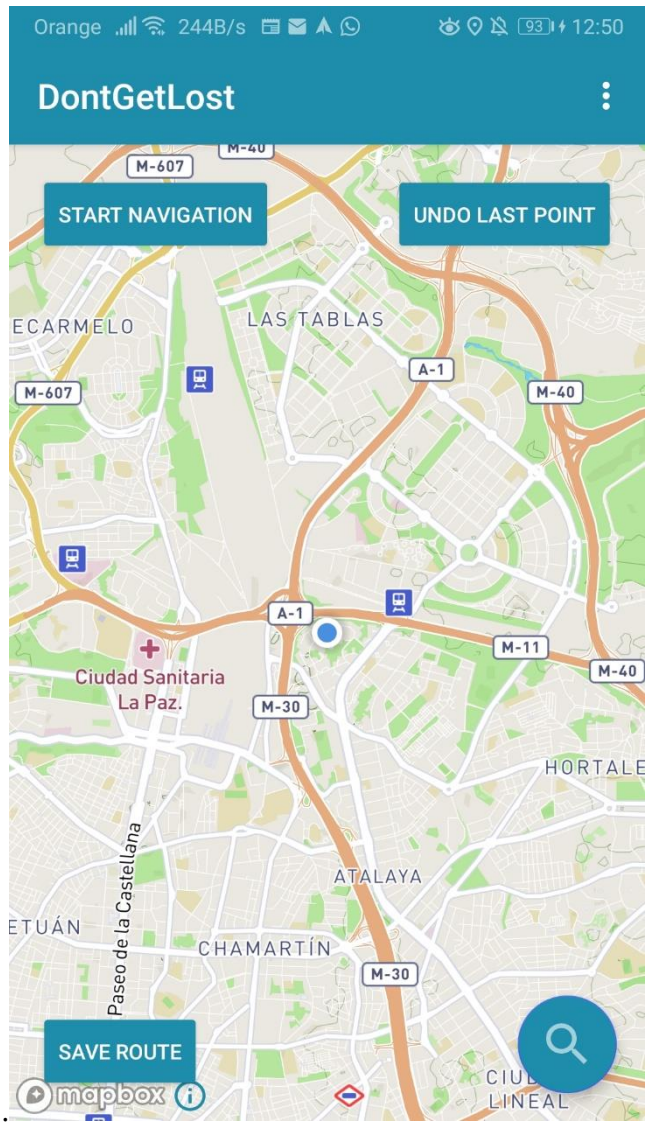

Figura 29: Prueba mapa sin ruta

En esta situación, se pueden realizar varias operaciones.

La primera prueba es la de utilizar la función táctil para situar un punto en el mapa.

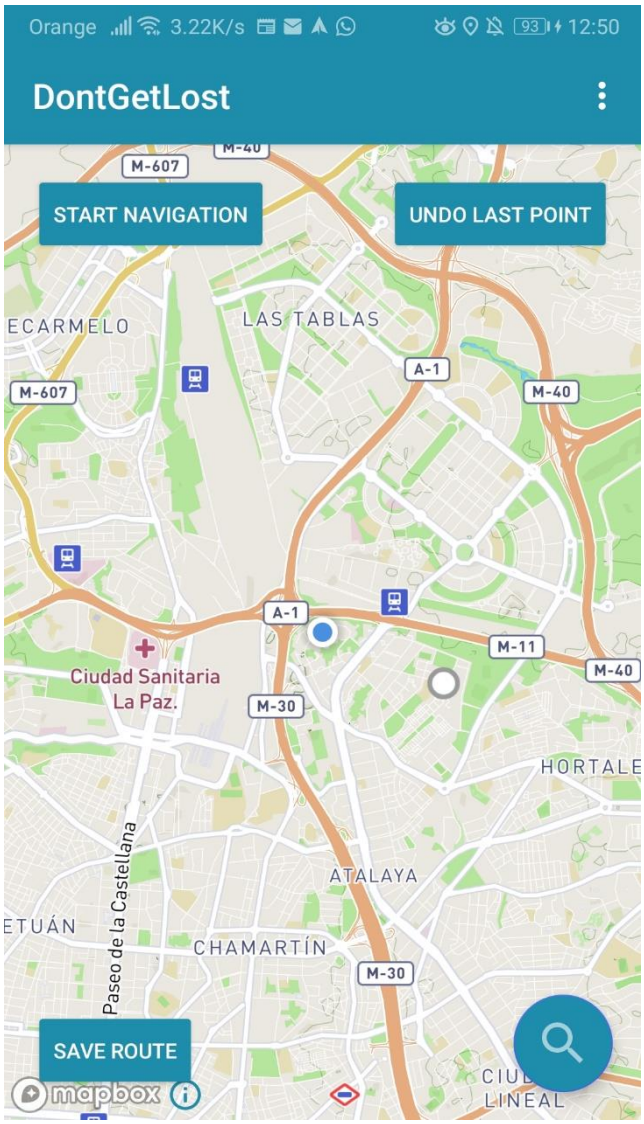

Figura 30: Prueba primer punto pulsando

Como podemos observar, la prueba es un éxito, ya que se nos muestra el punto de manera clara en el mapa.

Ya que una ruta de un solo punto no tiene mucho uso, nos disponemos a añadir más puntos.

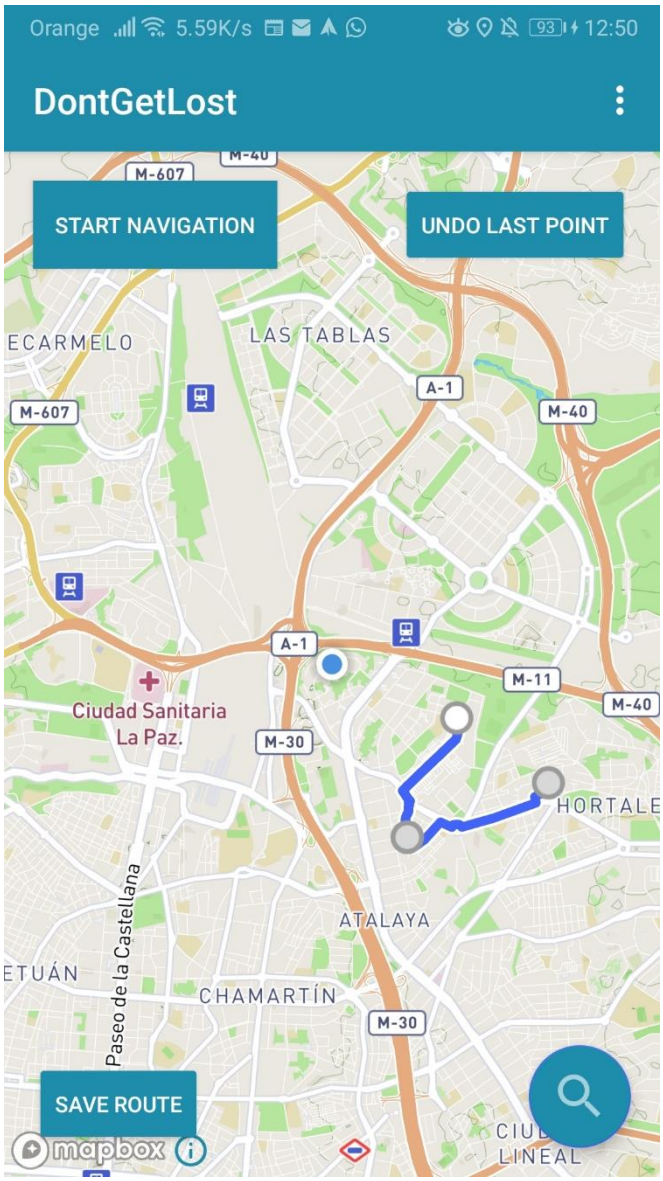

Figura 31: Prueba varios puntos

Con una ruta creada por varios puntos podemos ver cuál es el punto inicial (dibujado en blanco) y cuál es el camino que se seguirá.

Llegados a este punto, nos interesa guardar dicho circuito, para lo que pulsamos Guardar Ruta.

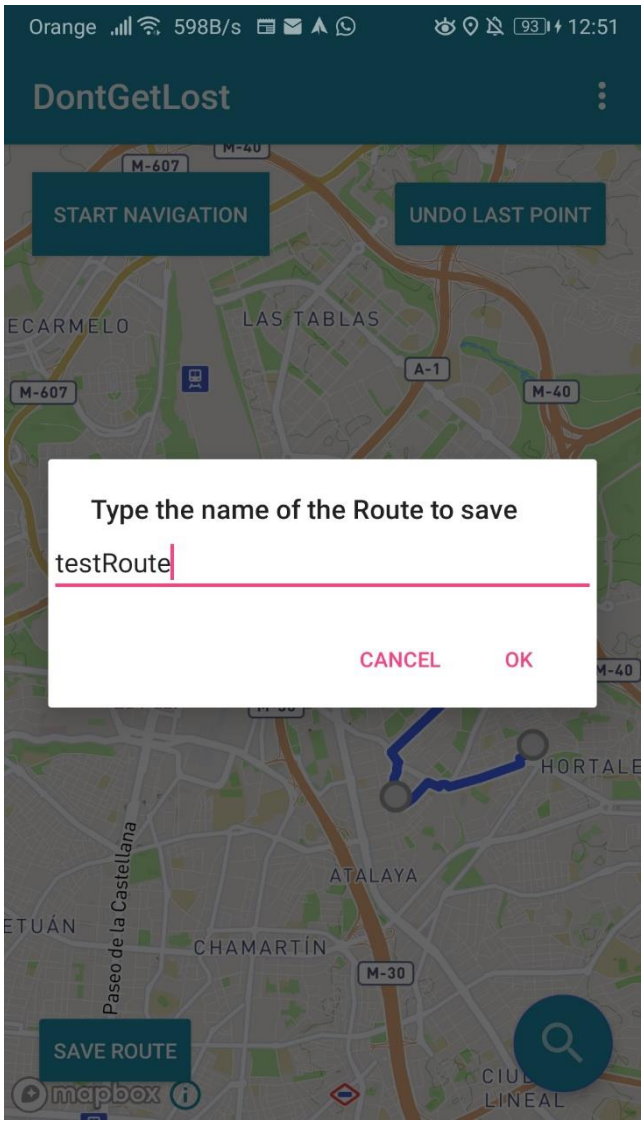

Figura 32: Prueba Guardar Ruta

Esta acción conlleva un cuadro de diálogo en el que se nos pide introducir el nombre de la ruta. Acabada esta acción, podremos volver al menú principal y comprobar que, efectivamente, se ha guardado la ruta.

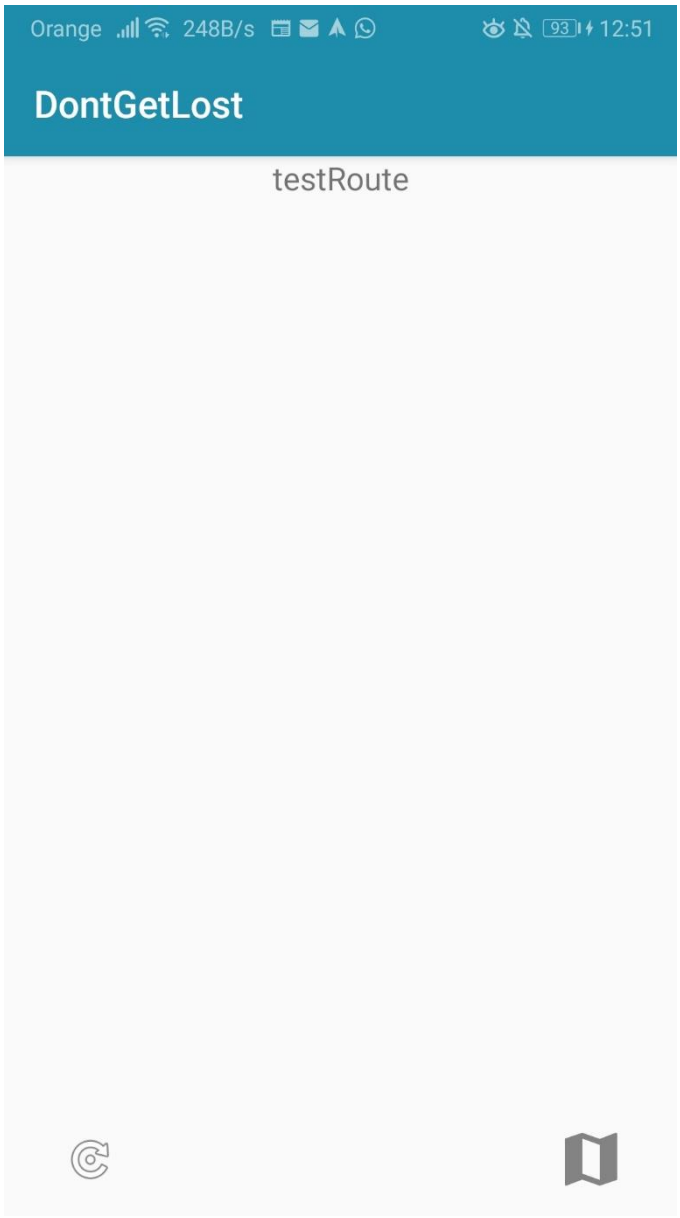

Figura 33: Prueba Menú principal

Efectivamente, se puede ver la ruta que hemos creado en el listado de rutas guardadas.

Las pruebas realizadas a continuación son dos, el cargado de la ruta en el mapa y el eliminado de la ruta.

Si se quiere cargar la ruta, se pulsa en el nombre de la ruta.

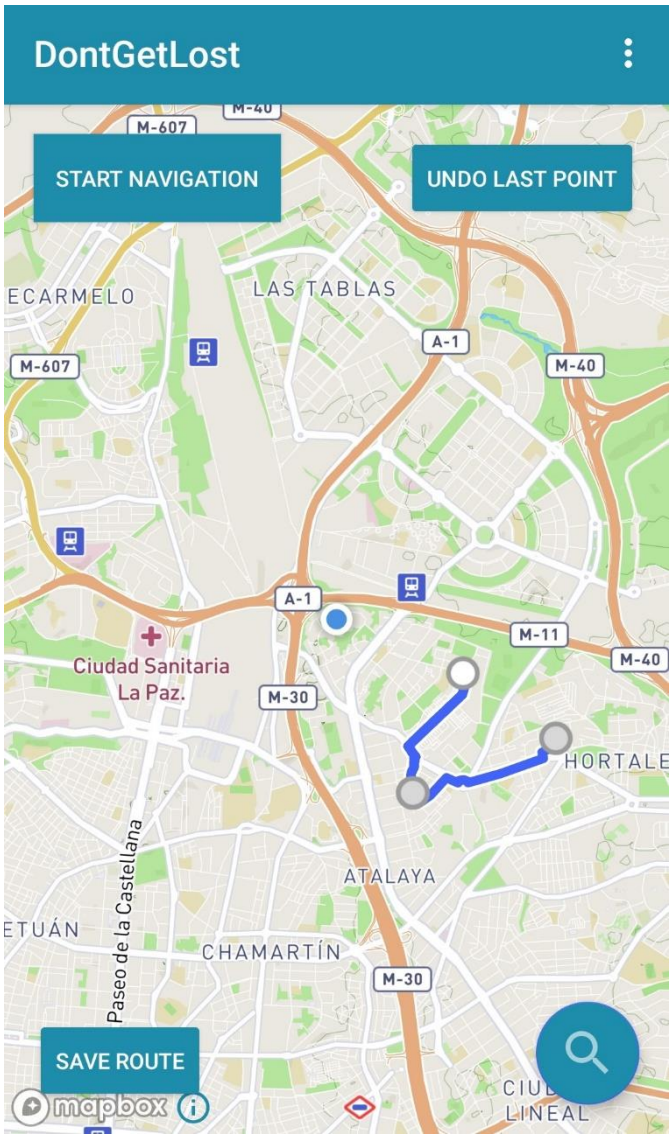

Figura 34: Prueba cargar ruta

Se puede comprobar que la ruta cargada es la misma que la creada con anterioridad, por lo que la prueba es un éxito.

Una vez cargada la ruta, podremos llamar al navegador para que nos guie en el circuito por los puntos hasta el destino. La aplicación internamente almacena tanto la información de la ruta actual creada como de la supuesta ruta que el usuario tomará cuando ejecute la opción de navegación. Esta ruta de usuario es la misma que la ruta actual, con la diferencia de que el origen es la localización del usuario, por lo que la aplicación nos guiará hasta el primer punto del circuito creado.

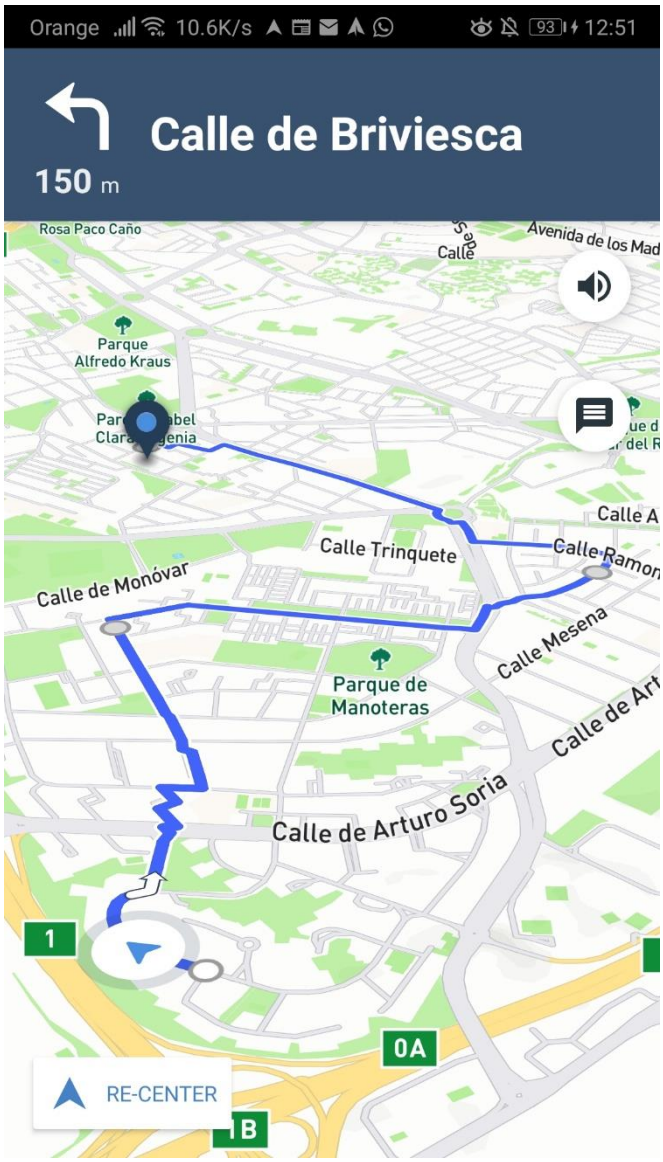

Figura 35: Prueba navegación ruta

Aquí podemos observar cómo el navegador detecta nuestra posición y realiza una navegación punto a punto desde el punto inicial de la ruta hasta el destino.

Tras finalizar la navegación, o en cualquier momento que se nos muestre el mapa con la ruta, podremos deshacer el último punto introducido pulsando el botón de deshacer.

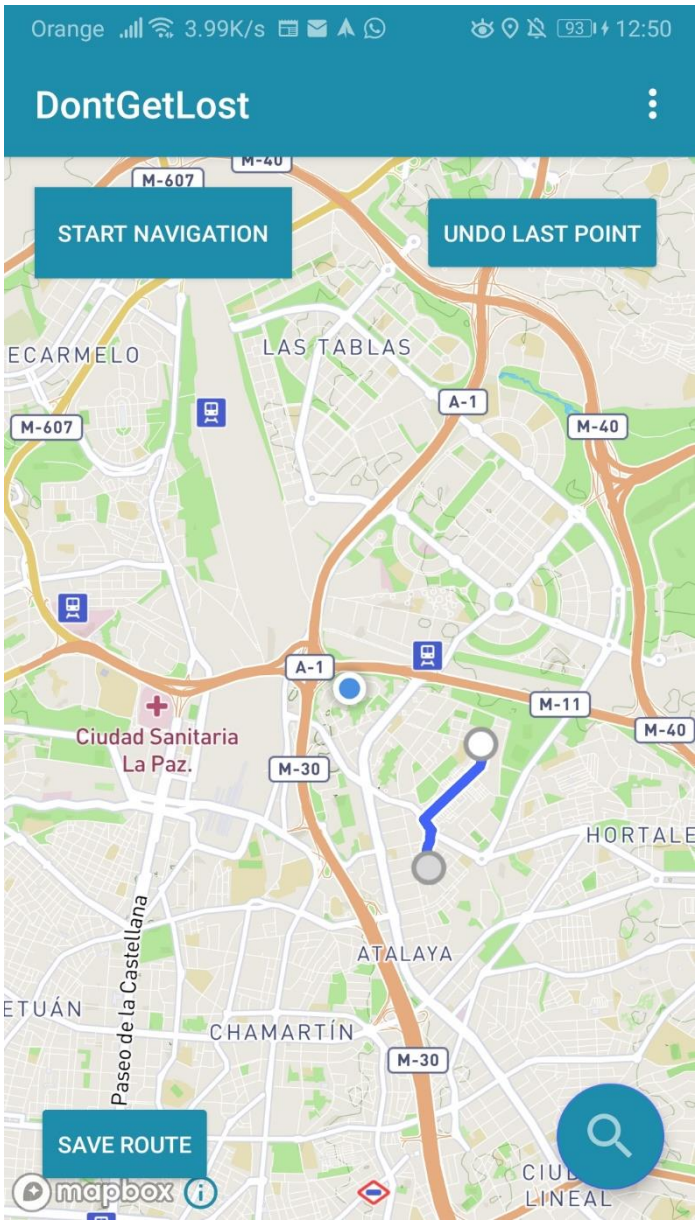

Figura 36: Prueba deshacer último punto

Como podemos observar, la ruta cambia al estado anterior y se elimina el último punto introducido.

Por último, se realizan las pruebas de la opción de búsqueda.

Si pulsamos el botón de la lupa situado en la parte inferior derecha del mapa, nos encontramos con esta interfaz.

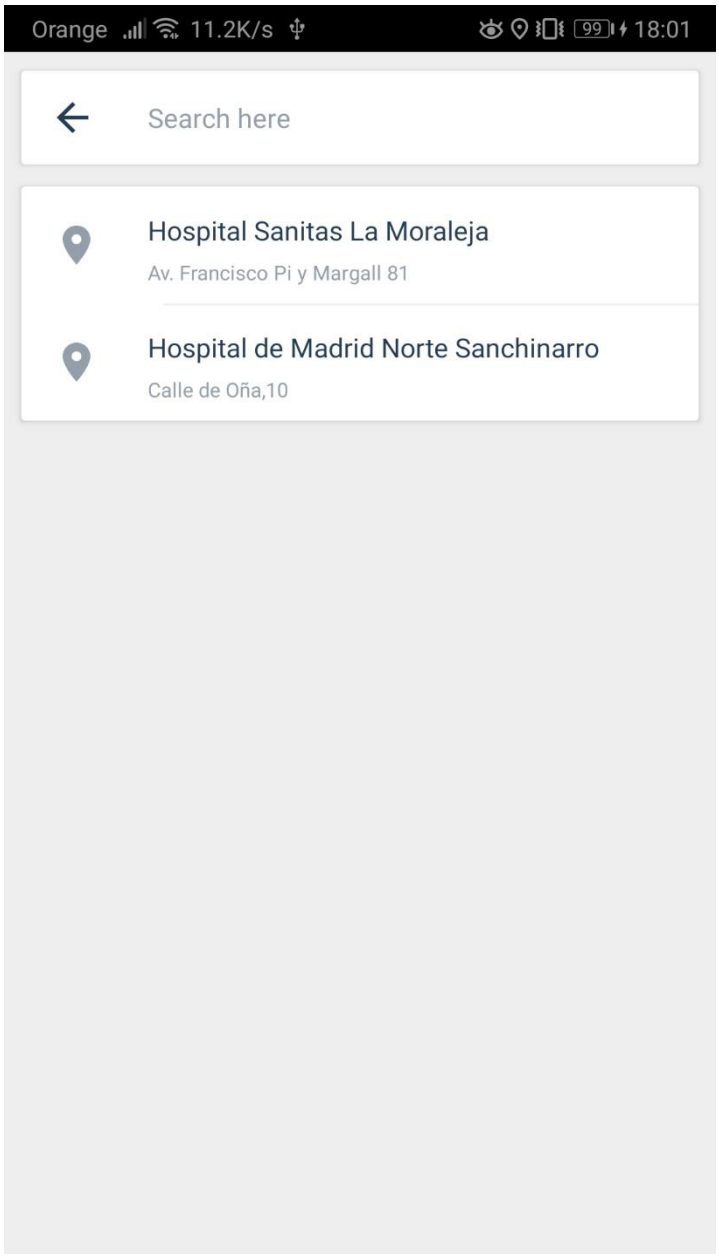

Figura 37: Prueba búsqueda

En él podremos buscar un punto de interés que queramos añadir en nuestra ruta introduciendo texto en el cuadro destinado a ello. Si se ha realizado alguna búsqueda con anterioridad, aparecerá como opción rápida en el historial de búsquedas que se encuentra debajo del cuadro. Si lo que queremos es buscar algo diferente, al introducir texto en el cuadro la aplicación irá mostrando los posibles resultados, de los cuales se seleccionará uno.

Tras pulsar el lugar deseado, la aplicación colocará un punto en dicho lugar y nos moverá la cámara hacia él para un mejor entendimiento de la localización del punto

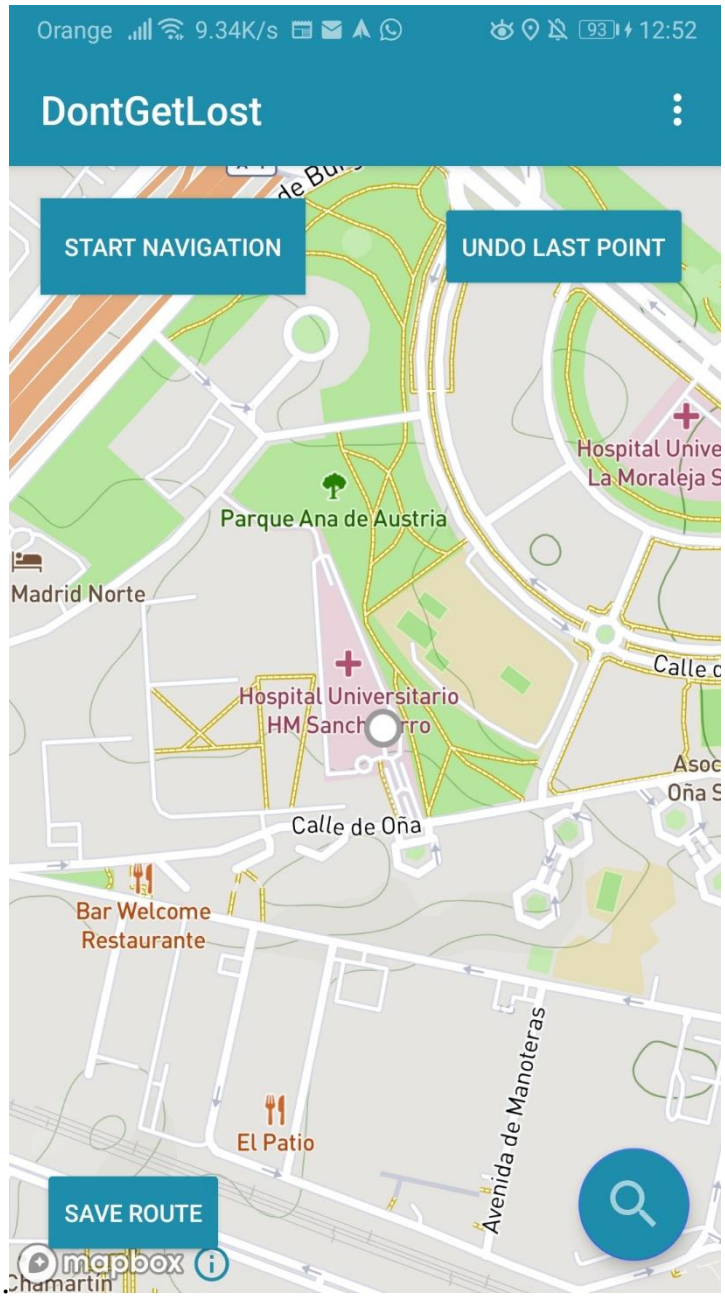

Figura 38: Prueba introducir punto de búsqueda

Si añadimos otro punto con la búsqueda o el sistema de pulsado táctil de pantalla, añadiremos el punto y se nos mostrará la nueva ruta.

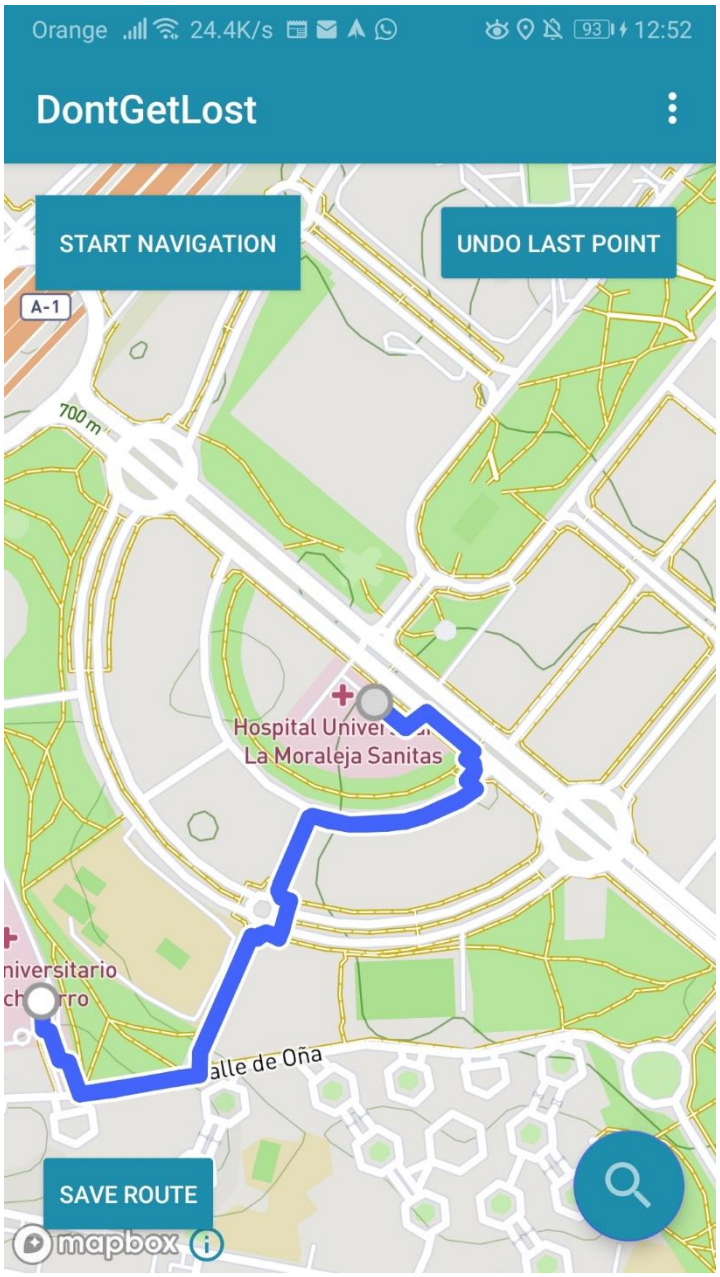

Figura 39: Prueba añadir punto tras búsqueda

Para la realización de la prueba de eliminado de ruta vamos a guardar esta ruta con el nombre de testSearchRoute.

Si volvemos al menú principal y realizamos un pulsado largo sobre el nombre, se mostrará un cuadro de diálogo para confirmar la eliminación.

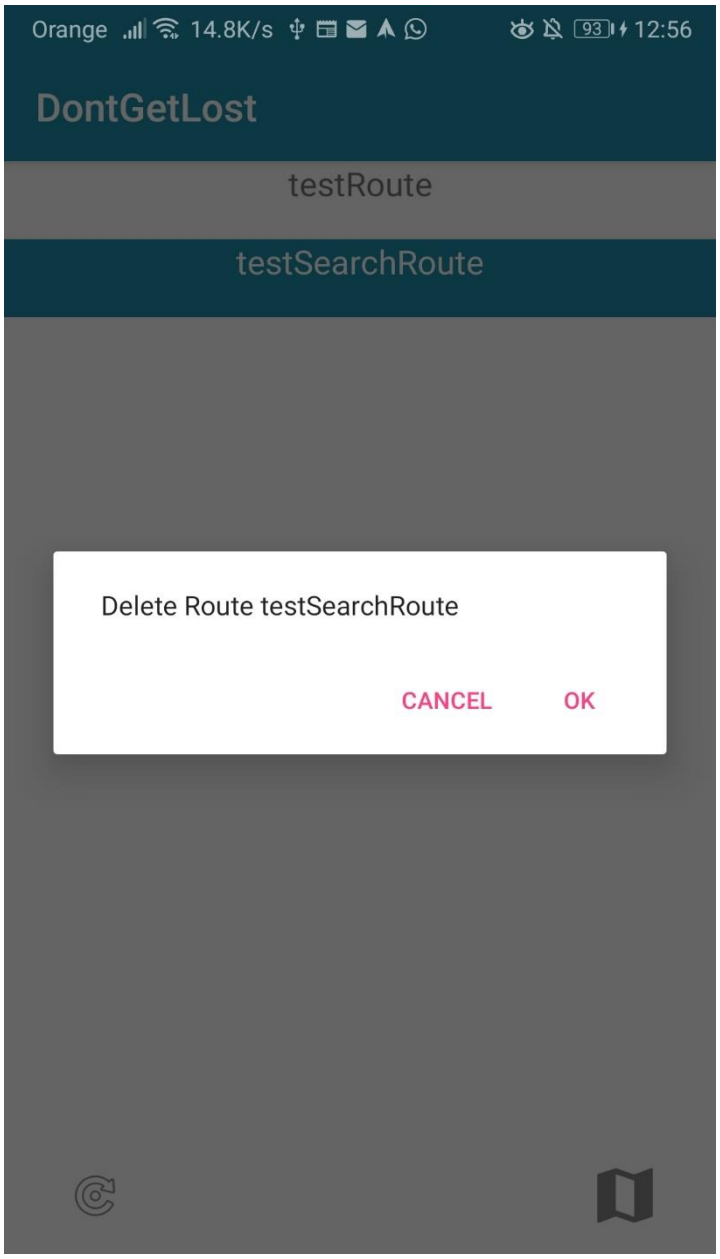

Figura 40: Prueba eliminar ruta

Tras lo cual, deberemos pulsar el botón de actualizar, situado en la parte inferior izquierda, para refrescar la lista de rutas, y se verá que se ha eliminado la ruta seleccionada.

Tras estas pruebas, se procede a realizar las propias con el módulo de puntos habituales.

Se comienza con la creación de un punto habitual.

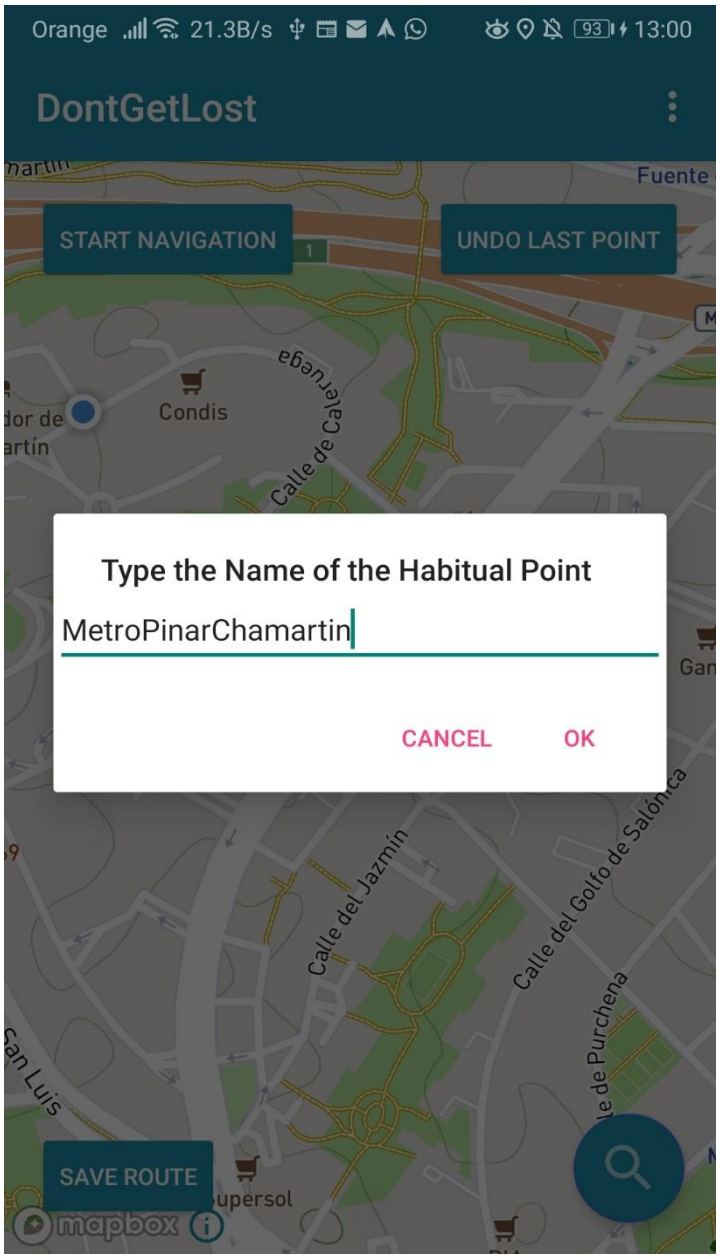

Figura 41: Prueba creación punto habitual

Se ha realizado una búsqueda de un punto y, al elegirlo, salta un cuadro de diálogo para introducir el nombre. Una vez confirmado, se podrá ver dicho punto en el listado de puntos habituales.

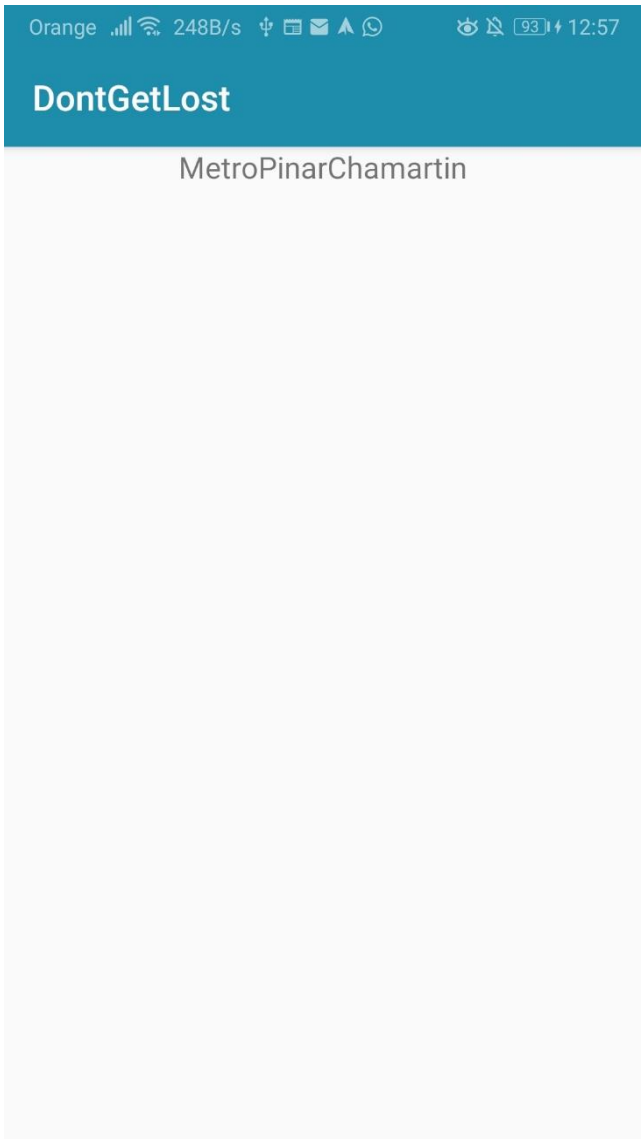

Figura 42: Prueba visualización punto habitual

Comprobadas la creación y visualización de puntos, se realiza la prueba de generación de una ruta con dichos puntos. Para ello se crea un segundo punto, el cual servirá de punto intermedio para nuestro circuito.

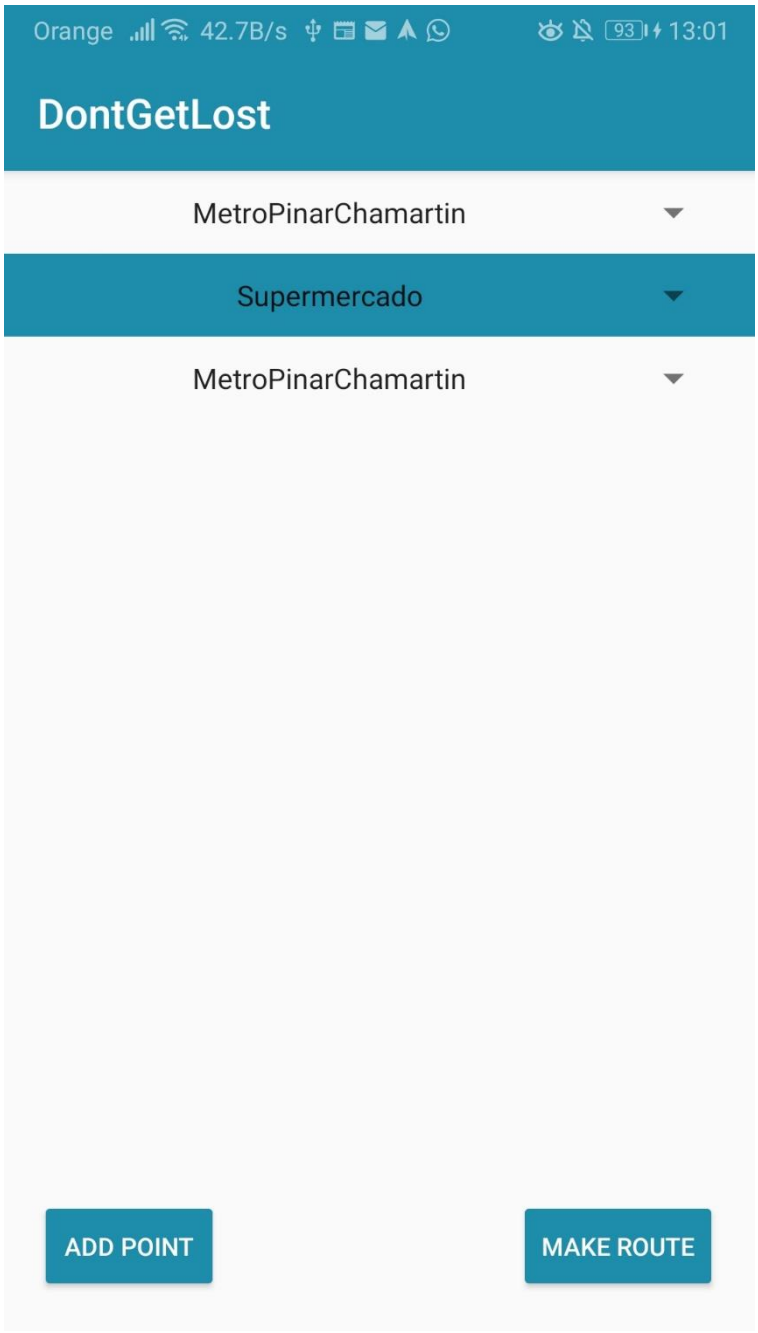

Figura 43: Prueba generación ruta puntos habituales

Para esta prueba se accede al menú de creación de ruta habitual. Se pulsa cuantas veces se quiera el botón de añadir punto y se escogen los puntos que formaran la ruta. En este caso es una ruta circular.

Al pulsar el botón de generar ruta, salta un mensaje de Ruta Guardada.

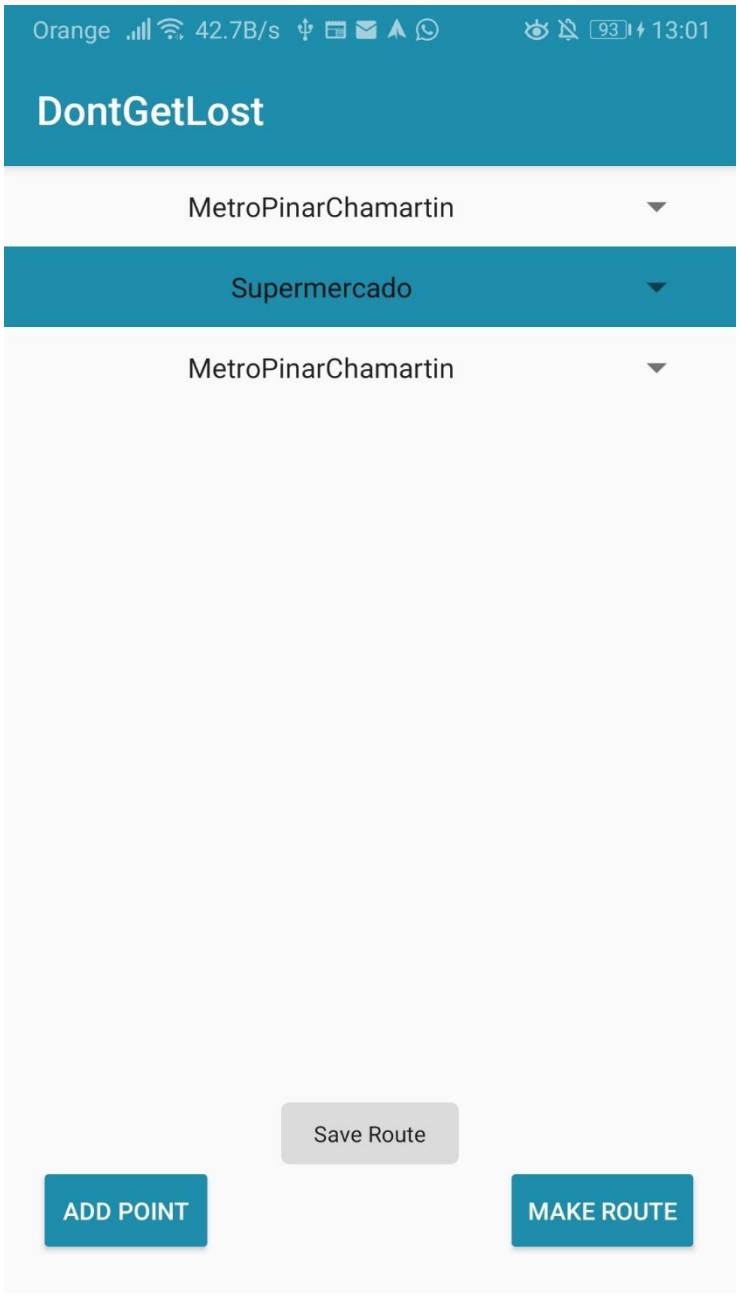

Figura 44: Prueba ruta habitual guardado

Si volvemos al menú principal, se podrá comprobar su creación.

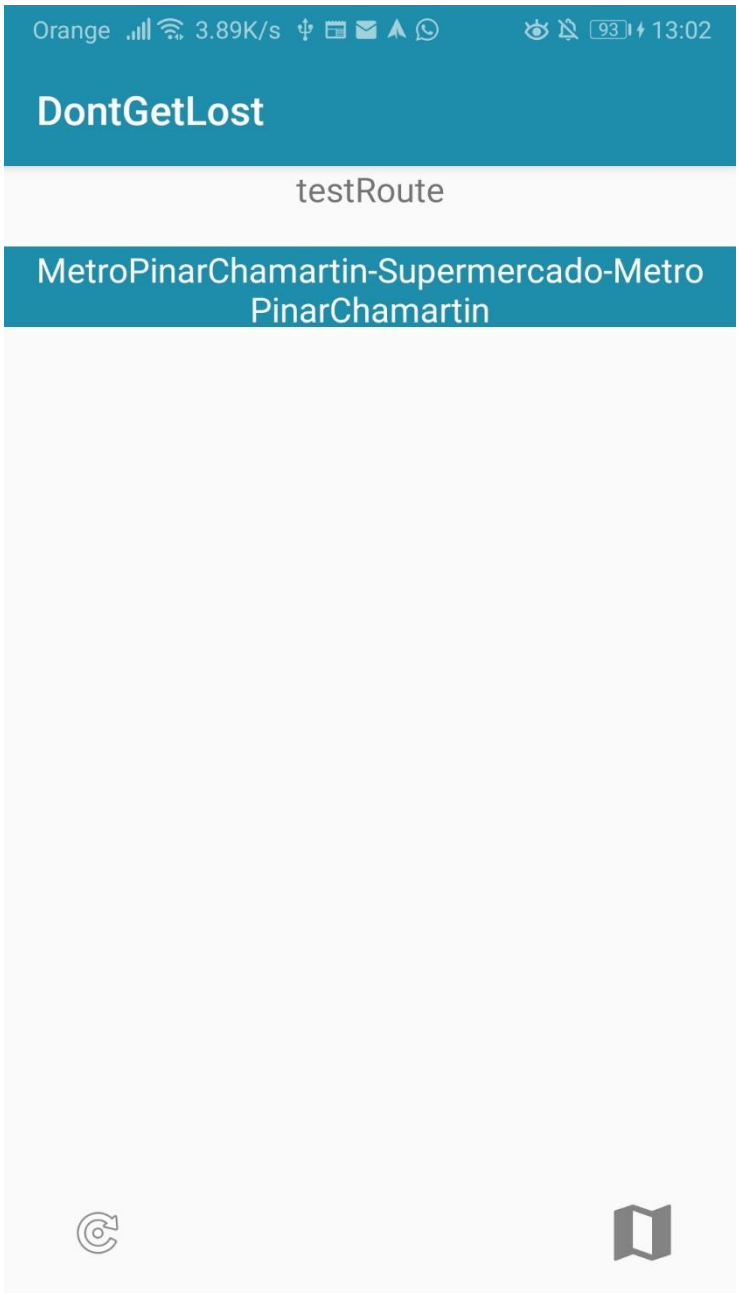

Figura 45: Prueba visualización ruta habitual

Tras lo cual, se podrá pulsar y ver en el mapa.

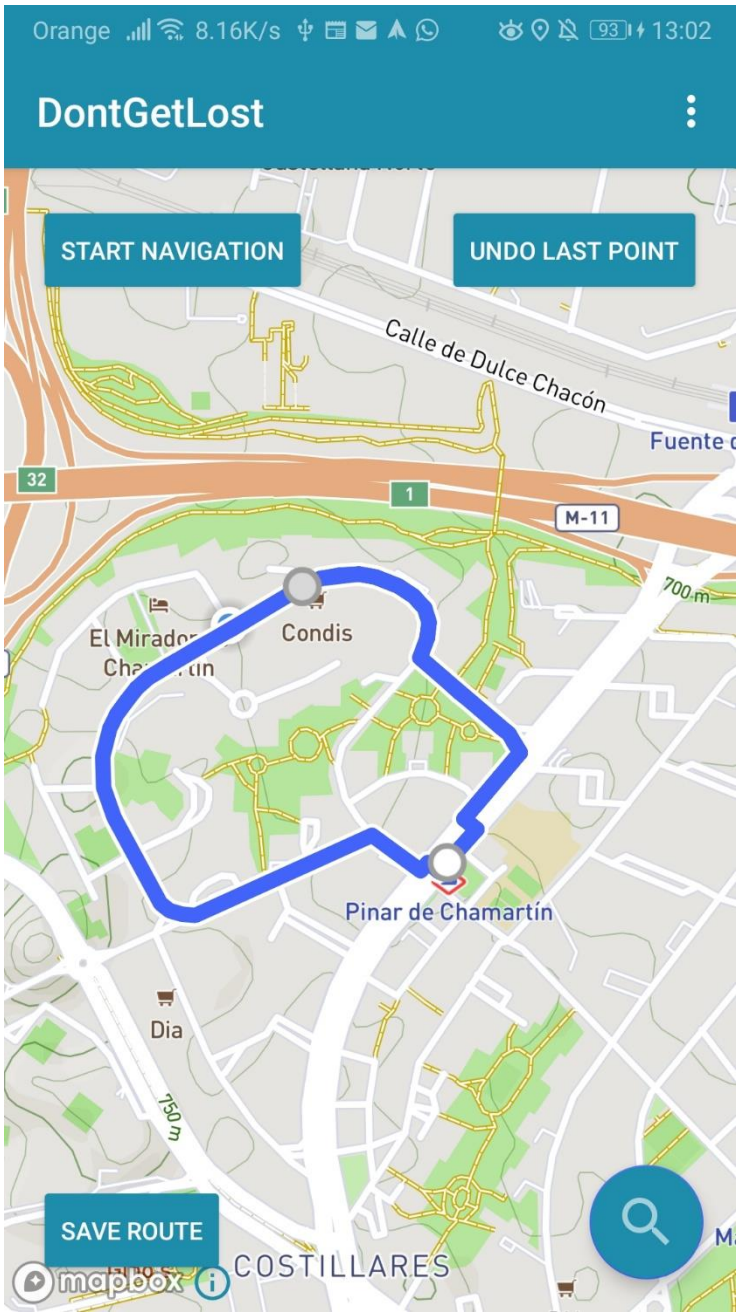

Figura 46: Prueba ruta habitual en mapa

Previamente a la creación de más puntos, se realizó la prueba de eliminación de un punto habitual.

El proceso es similar al de las rutas. Pulsado largo en el punto, tras lo cual salta un cuadro de confirmación.

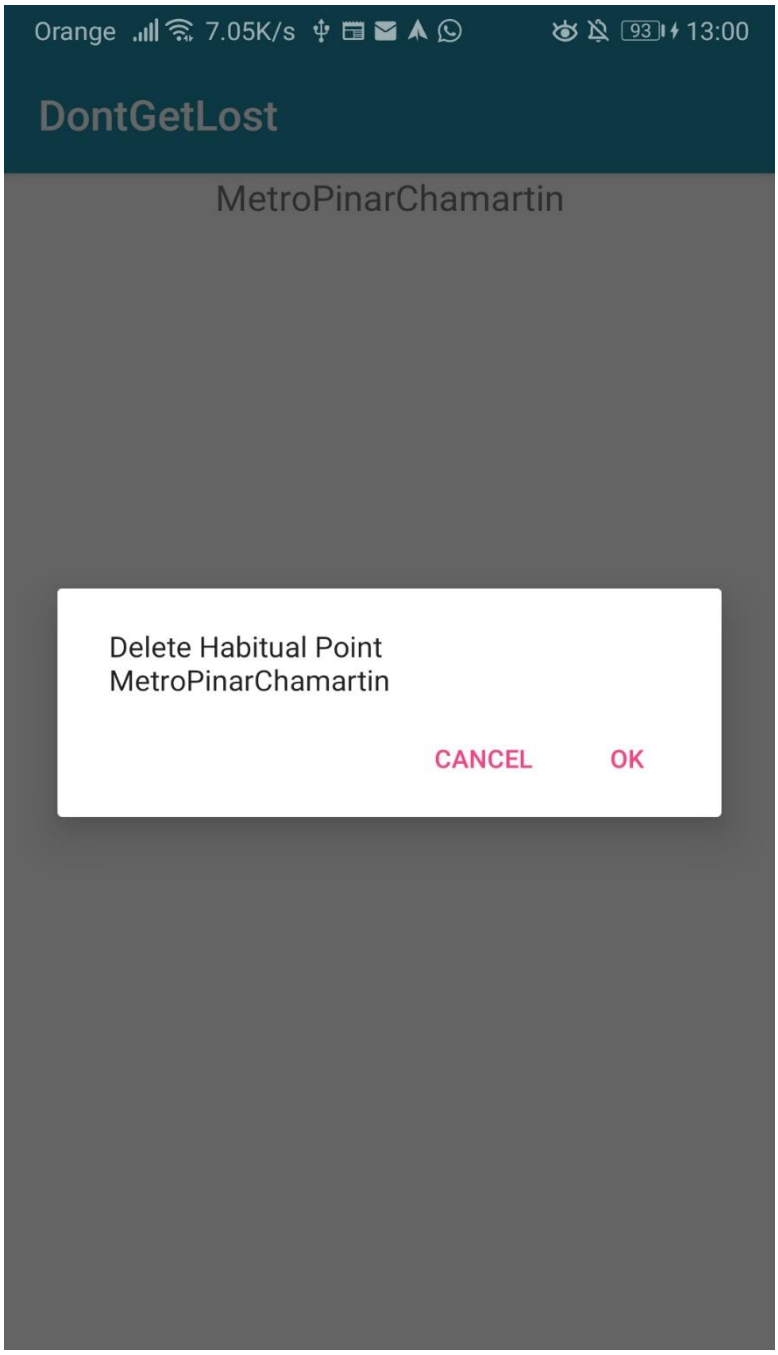

Figura 47: Prueba eliminación punto habitual

Por último, se realizaron unas pruebas de comprobación de errores.

Estas son:

Intentar deshacer un punto en el mapa cuando no hay más puntos que deshacer: en esta prueba se comprueba que el usuario recibe un mensaje que no se puede deshacer ningún punto más.

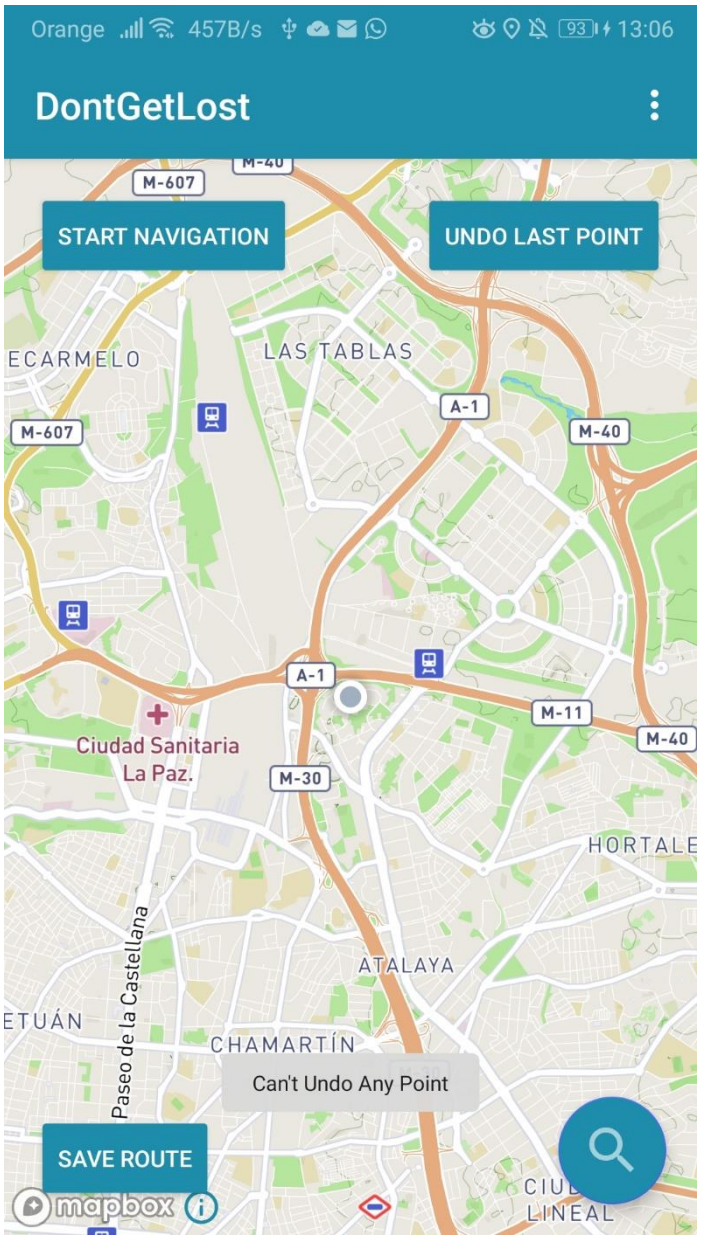

Figura 48: Prueba error deshacer punto

Intentar navegar una ruta cuando no se ha agregado ningún punto. La aplicación mostrará un mensaje de error al no poder navegar una ruta que no existe.

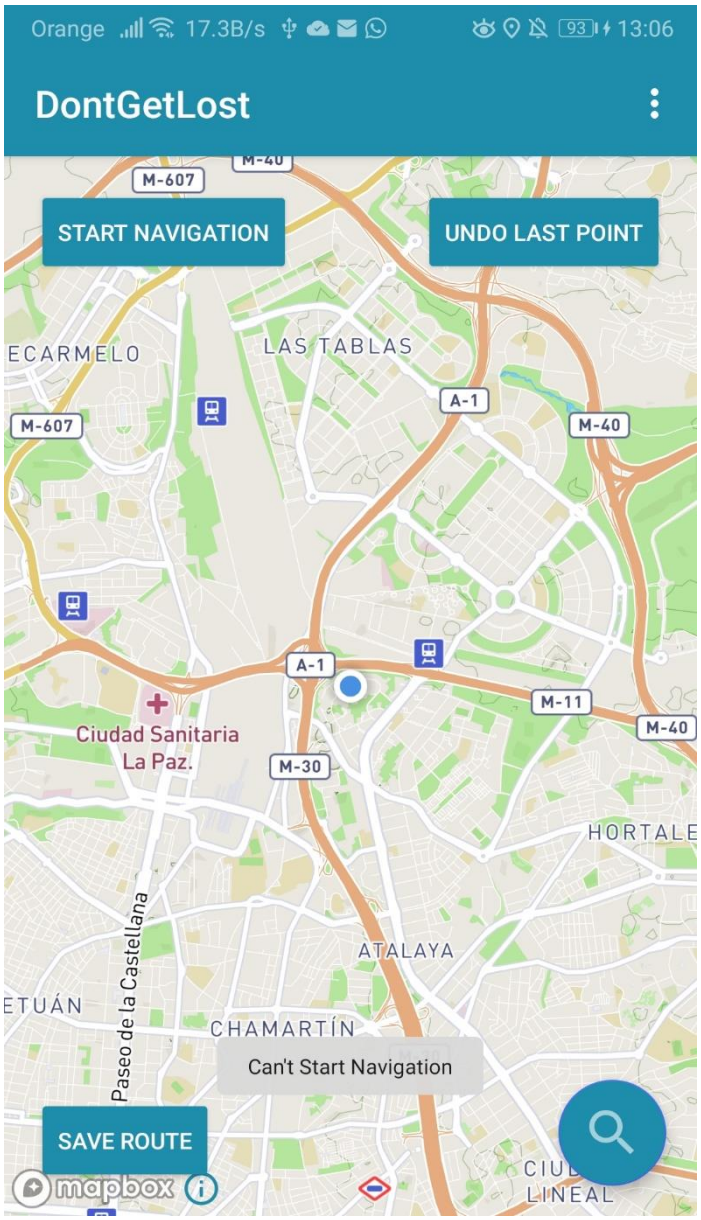

Figura 49: Prueba navegación error

E intentar guardar una ruta cuando no se ha agregado ningún punto. La aplicación mostrará un mensaje de error que indica que no se puede guardar una ruta que no existe.

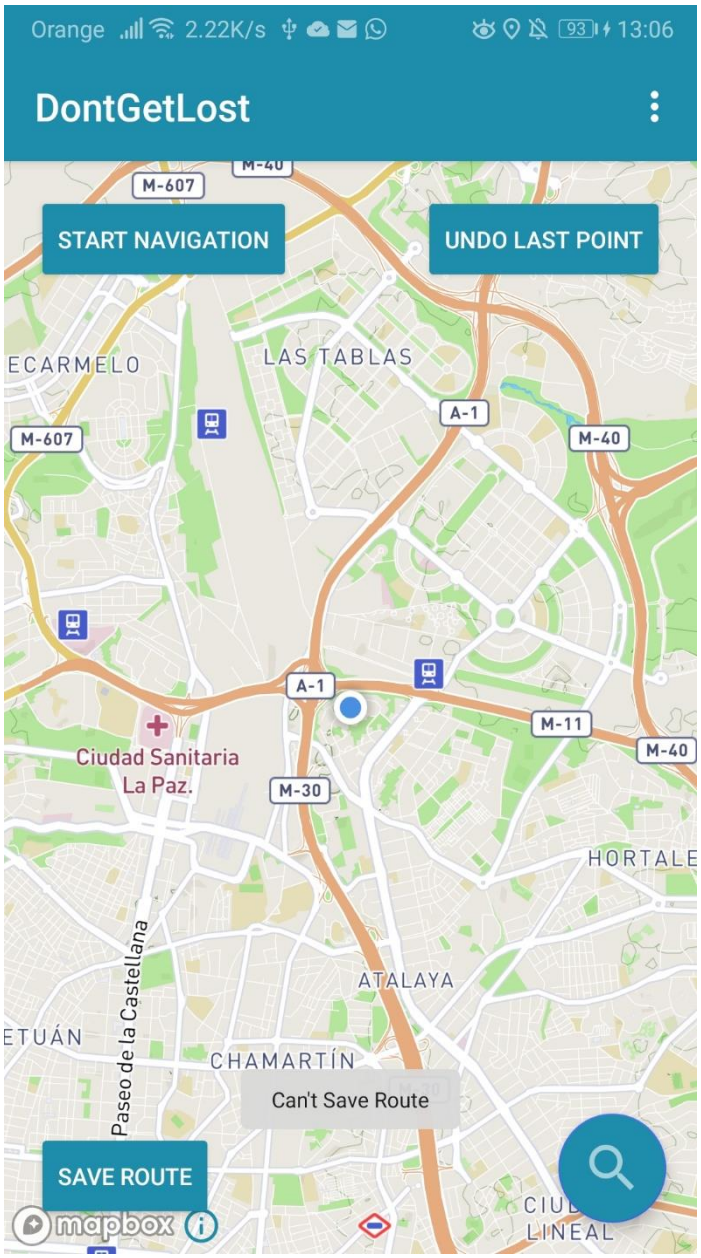

Figura 50: Prueba guardado ruta error

6 Conclusiones y trabajo futuro

6.1 Conclusiones

Llegados al apartado de conclusiones es de importancia repasar si se ha cumplido con los objetivos previstos.

El objetivo claro y primordial por el cual se está realizando esta memoria es el desarrollo de una aplicación que permita el diseño y la creación de rutas en un entorno GPS que posteriormente pueda ser utilizado para guiar al usuario de punto a punto, recorriendo el itinerario multipunto hasta el final. Este objetivo claramente se ha superado con la existencia de un software que realiza estas tareas de forma eficiente.

Era importante que la aplicación contara con una sencillez de uso notable, ya que es importante que este software pueda ser utilizado por cualquier persona, incluso aquellas con pocos conocimientos sobre aplicaciones móviles.

En cuanto al diseño de las rutas, era importante dar diferentes posibilidades a la hora de añadir puntos al itinerario. Esto se ha conseguido introduciendo dos métodos: la posibilidad de pulsar en el mapa para añadir ese punto y el hecho de poder buscar un lugar de interés con un buscador propiamente desarrollado.

Por supuesto, se puso como objetivo el poder guardar estas rutas que se están diseñando para poder ser utilizadas más adelante. Esto se ha desarrollado de forma correcta en la aplicación. Una vez guardadas, es posible utilizar estas rutas en futuras ocasiones y realizar una navegación precisa y rápida por los puntos de la ruta, hasta el último.

También se ha dado la posibilidad de crear y gestionar puntos y rutas habituales. Esta función es muy útil si se pretende utilizar una serie de puntos con mucha frecuencia. La creación de puntos es sencilla e intuitiva, y su uso en la generación de rutas habituales es muy cómoda, con lo que se pueden crear de manera rápida y eficaz. Estas operaciones permiten utilizar rutas habituales más rápidamente y dar más flexibilidad a la aplicación para la gestión de diferentes tipos de rutas, ya sean personalizadas o habituales.

Como conclusión, se tiene una aplicación móvil que es sencilla en su uso y potente en sus funcionalidades, por lo que se considera satisfactorio el desarrollo de esta aplicación, ya que

a su vez tenemos un software modular, fácilmente ampliable en sus funciones y que se puede tomar como punto de partida para otras ideas y aplicaciones.

6.2 Trabajo futuro

Como es de esperar en toda aplicación, se pueden realizar ampliaciones de sus funciones. En este caso, se han considerado varias posibilidades francamente interesantes.

Si se desea una orientación más social, es posible desarrollar un sistema de avisos en momentos en los que el usuario se separe de la ruta, para evitar que se pierda, y, en caso de no volver a la ruta, se podría notificar a una tercera persona avisando de los inconvenientes que el usuario está teniendo.

Una característica que actualmente está muy de moda y de la que podría beneficiarse la aplicación es la introducción de un sistema de aprendizaje automático que aprenda del usuario durante un periodo de tiempo tras el cual el software sugiera rutas multipunto que se añadan a las rutas habituales para su posterior uso.

Combinando ambas funciones, se podría crear una aplicación francamente útil en el ámbito social y con una amplia aplicación en personas mayores o con problemas de orientación.

Referencias

[1] Google Maps: https://www.google.com/maps. Último acceso: 23/04/2020

[2] Sygic: https://www.sygic.com/. Último acceso: 23/04/2020

[3] TomTom: https://www.tomtom.com/. Último acceso: 23/04/2020

[4] View Ranger: https://www.viewranger.com/. Último acceso: 23/04/2020

[5] Mapbox: https://www.mapbox.com/maps/. Último acceso: 23/04/2020

[6] Mapbox documentación: https://docs.mapbox.com/android/maps/overview/. Último acceso: 23/04/2020

[7] Android Studio: https://developer.android.com/studio. Último acceso: 23/04/2020

[8] UML documentación: https://www.omg.org/spec/UML/2.5.1/PDF. Último acceso: 23/04/2020

[9] Draw.io: https://draw.io. Último acceso: 23/04/2020

[10] Java: https://www.java.com/. Último acceso: 23/04/2020

[11] Java documentación: https://docs.oracle.com/javase/7/docs/api/. Último acceso: 23/04/2020

Glosario

API	APPlication Programming Interface
UML	Uniform Resource Locator
APK	Android Package Kit
GPS	Global Positioning System
UML	Unified Modeling Language
IDE	Integrated Development Environment
XML	Extensible Markup Language
GUI	Graphical User Interface

Anexos

A *Manual de instalación*

Al tratarse de una aplicación Android, el proceso de instalación que he elegido es la distribución de un archivo APK a través de un repositorio propio creado en GitLab.

Para la descarga de dicho archivo es necesario entrar en la URL:

https://gitlab.com/javierdemarco/dontgetlostapk/-/tree/master/release

Y descargar el archivo dontgetlost.apk con un dispositivo Android.

Una vez realizada esta tarea, navegar a la carpeta de descargas y presionar en el archivo APK. Como es una aplicación externa a la Google Play Store, el dispositivo mostrará un aviso de si confiamos en el origen de este archivo a lo que pulsaremos sí.

El instalador del dispositivo se encargará posteriormente de instalar la aplicación, tras lo cual podremos lanzar la APP y utilizar todas sus funciones.